DATA-DRIVEN APPROACH USING MACHINE LEARNING FOR REAL-TIME FLIGHT PATH OPTIMIZATION

A Dissertation
Presented to
The Academic Faculty

By

Junghyun Kim

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Computational Science and Engineering
(Home Unit: Aerospace Engineering)

Georgia Institute of Technology

May 2021

DATA-DRIVEN APPROACH USING MACHINE LEARNING FOR REAL-TIME FLIGHT PATH OPTIMIZATION

Thesis committee:

Prof. Dimitri N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Prof. Daniel P. Schrage
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Simon I. Briceno
Chief Commercial Officer
Jaunt Air Mobility

Prof. Polo Chau
School of Computational Science and Engineering
Georgia Institute of Technology

Prof. Chao Zhang
School of Computational Science and Engineering
Georgia Institute of Technology

Date approved: April 23, 2021

So, whether you eat or drink, or whatever you do, do all to the glory of God.

1 Corinthians 10:31

For the glory of God

ACKNOWLEDGMENTS

As I am now rounding the last corner before the end of my academic journey, I would like to take a few moments to sincerely thank all of those who have supported me in numerous different ways. Honestly, as there are so many people to list, this section has been one of the hardest parts of this dissertation for me. This journey would have not been possible without the dedication and support of my advisor, committee members, colleagues, friends, mentors, and family.

First of all, I would like to express my sincere gratitude to my advisor, Prof. Dimitri Mavris. He has been an excellent mentor since I joined the Aerospace Systems Design Laboratory (ASDL) as a Ph.D. student in Fall 2015. I am grateful for the opportunities he has provided me - for teaching me the process of research, for recruiting me as a graduate research/teaching assistant, for allowing me to have summer internship experiences, for making me participate in various research projects, and, most importantly, for trusting me to complete this dissertation. I would not have succeeded in completing my academic goals without his continuous guidance and support. I could not have asked for a more supportive advisor in my life. I have been truly blessed to work with the best advisor. Thanks Doc for your guidance, patience, encouragement, and support. I will never forget what you have done for me.

I would also like to express my profound appreciation to my committee members: Prof. Chao Zhang, Prof. Daniel Schrage, Prof. Polo Chau, and Dr. Simon Briceno. They have been tremendously supportive by providing me a multitude of invaluable insights and feedback leading to the completion of a well-rounded dissertation. I believe their perspectives will shape my research career for years to come. Thanks to all of you for providing world-class guidance and taking the time to review my work. I would especially like to extend a special thanks to Dr. Simon Briceno who helped me formulate the ideas, placed my work in the right context, and, most importantly, spent numerous hours meeting

with me over the last few years.

During my time in the ASDL, I have been blessed with the opportunity to work with many incredibly talented people: Airline Operations Research Group, who is Coline Ramee, Marie Deguignet, Saeyone Balasekar, Anh-Thu Nguyen, Dr. Youngjun Choi, and Dr. Cedric Justin; Aviation Environmental Research Group, who is Ameya Behere, Zhenyu Gao, Eva Kallou, Fatma Karagoz, Dylan Monteiro, Dr. Yongchang Li, Dr. Dongwook Lim, Dr. Holger Pfaender, Dr. Mohammed Hassan, and Dr. Michelle Kirby; ASDL Teaching Assistant Group, who is Melek Ozcan, Luis Nunez, Salah Tarazi, Efe Yarbasi, and Dr. Elena Garcia. I would particularly like to offer my sincere thanks to Dr. Cedric Justin who provided constructive feedback on my work. Also, thanks to Tanya and Adrienne for providing a significant amount of administrative assistance to me.

Having been at the Georgia Institute of Technology for five years and nine months, I have also found myself deeply humbled by many meaningful friendships I have formed. I would like to express my warmest thanks to my friends: Edwin Goh, Jingdao Chen, Seulki Kim, Sungil Hong, Soobum Kim, Hoon Na, Kisun Song, Kyuman Lee, Jinwoo Go, Takehiko Hanada, Chicka Hanada, Roddick Yu, Yuanlai Zhou, Hua Li, Uthaipon Tao, Jerry Kuo, Amy Liu, Gustav Davidsson, Esther Ling, Melvin Matthew, Kreston Barron, James Pagan, Thomas Lin, Tim Chin, Vincent Lee, Chidiebere Okoli, and David Caro. I would especially like to extend my appreciation to Rob Griffin, Karen Peterson, Kendra Slayton, and Kurt Belgum for the many discussions to improve my English skills.

Outside of my graduate studies, I am especially grateful to my spiritual brothers and sisters in Christ, many of whom I am now honored to count as my friends in Atlanta: Jesse Brannen, Delaine Brannen, Aaron Menikoff, Bryan Pillsbury, Tracey West, Suzanne West, Austin Slater, Jessi Slater, Soonhyong Seth Kwon, Youngmin Koh, Lloyd Seok, Bart Glasgow, Coley Glasgow, Joe Mercer, Angie Mercer, Lyndon Akins, Terry Akins, Brian Boham, Joann Boham, Nancy Bulk, Donald Bulk, Jay Duff, Chuck Emerson, Debbie Emerson, Neale Hightower, Carol Hightower, and Judy Billings. They have taught me the

power of faith and prayer. I am also thankful to Prof. Jon Ahn, Prof. Kyungtae Lee, and Prof. Kyuhong Kim for mentoring me during my academic journey. I am completely indebted to you all.

I want to thank my family for their unconditional love and unwavering support. First and foremost, I would like to express my deepest affection to my wife, Shinae Seo. She has walked with me on this long path and shared in all of its joys and sorrows. Thank you for your love, patience, support throughout my graduate studies, and, most importantly, thanks for accepting me for who I am. I love you with all my heart and look forward to our future in Texas and Virginia together. I would also like to extend my deepest gratitude to my parents, Yeungsung Kim and Yeoungrye Choi, who have supported me tirelessly and kept praying for me constantly throughout my life. My older sister, Yumi Kim, also provided me with the best example of how to be successful as a student abroad. I am also thankful to my parents-in-law, Jongwon Seo and Haeyeon Lim, for their encouragement and prayers. Ultimately, I could not have made this journey without the love and support of my family.

Last but not least, my dearest gratitude goes to my savior, Jesus Christ, to whom this dissertation is dedicated. Thanks to God for sending me to the United States and helping me realize that you are the only one who can control everything in my life. I give all that I am and ever hope to be to you, who is the giver of all good gifts.

TABLE OF CONTENTS

Acknowledgments v

List of Tables xii

List of Figures xiii

List of Acronyms xviii

Summary xxii

Chapter 1: Introduction 1

1.1 Research Motivation 1

1.2 Research Objective 8

1.3 Research Scope 9

1.4 Dissertation Structure 10

Chapter 2: Literature Review 12

2.1 Flight Management System 12

2.2 Ground-based Flight Planning Framework 13

2.3 Electronic Flight Bag 16

2.4 Machine Learning-based Flight Route Prediction 20

2.5 Research Gap 22

Chapter 3: Background 27

3.1 Artificial Neural Network 27

3.2 Support Vector Machine 34

3.3 Gaussian Process 36

3.4 Density-Based Spatial Clustering of Applications with Noise 36

3.5 K-Nearest Neighbor 39

3.6 A* Search Algorithm 42

Chapter 4: Research Formulation 47

4.1 Supervised Machine Learning-based Wind Regression 49

4.2 Unsupervised Machine Learning-based Convective Weather Prediction . . . 54

4.3 Designated Points-based Flight Path Optimization 60

4.4 Summary of Research Statements 64

Chapter 5: Implementation 70

5.1 Data Preparation 73

5.1.1 Global Forecast System (GFS) 73

5.1.2 Next Generation Radar (NEXRAD) 75

5.1.3 Convective Significant Meteorological Information (SIGMET) . . . 77

5.1.4 Meteorological Aerodrome Report (METAR) 78

5.1.5 Pilot Report (PIREP) 79

5.1.6 Flight Tracking Data 80

5.1.7 Air Traffic Service (ATS) Route and Waypoint 80

5.2 Enabler 1: Supervised Machine Learning-based Wind Model 82

5.2.1 Spatial Regression 83
5.2.2 Temporal Interpolation 87
5.2.3 Research Experiment 89
5.3 Enabler 2: Unsupervised Machine Learning-based Convective Weather Model 96
5.3.1 Data Collection 97
5.3.2 Data Filtering 97
5.3.3 Data Clustering 98
5.3.4 Polygon Generation 99
5.3.5 Research Experiment 100
5.4 Enabler 3: Designated Points-based Flight Path Optimization Model 103
5.4.1 Assumptions 104
5.4.2 Airspace Network Generation 104
5.4.3 Bounding Box 105
5.4.4 Travel Time Estimation 107
5.4.5 Collision Check 109
5.4.6 Route Generation 112
5.4.7 Research Experiment 117
5.5 Software Architecture 122
5.5.1 Data Pre-processing 123
5.5.2 Data Post-processing 124

Chapter 6: Results and Discussion 126

6.1 Statistical Analysis 126

6.1.1 Flight Data Collection . . . 126
6.1.2 Flight Trajectory Clustering . . . 128
6.1.3 Representative Flight Selection . . . 130
6.1.4 Verification and Validation . . . 133
6.1.5 Assessment of Potential Benefits . . . 136
6.2 Runtime Analysis . . . 143

Chapter 7: Conclusion . . . 147
7.1 Recapitulation of Research Statements . . . 147
7.2 Summary of Contributions . . . 150
7.3 Potential Applications . . . 151
7.4 Recommendations for Future Work . . . 152

Appendices . . . 154
Appendix A: Pilot Interview Transcription . . . 155
Appendix B: Wind Measurement Data by Weather Balloons . . . 162
Appendix C: NEXRAD Site Information (Contiguous United States) . . . 165

References . . . 171

Vita . . . 181

LIST OF TABLES

1.1 Total cost of delay (unit: $billion) 2

2.1 ASDL-initiated flight path optimization software 23

4.1 The most well-known numerical weather models 49

5.1 Relationship between radar reflectivity and intensity of rainfall 76

5.2 Design of Experiment results for the hyperparameters of the MLP model . . 85

5.3 Supervised machine learning-based wind regression model evaluation . . . 90

5.4 Supervised machine learning-based regression vs. Linear interpolation . . . 92

5.5 Travel time and Hausdorff distance comparison between DL971 and RTOP 120

6.1 Travel time and Hausdorff distance comparison between DL1072 and RTOP 140

B.1 Wind measurement data by the NOAA's weather balloons at 2021-02-02 12:00 UTC (Altitude = 250 hPa) 162

C.1 NEXRAD site location information within contiguous U.S. 165

LIST OF FIGURES

1.1 Total number of flights tracked by Flightradar24 (Adapted from [5]) 2

1.2 Sources of flight delays in the U.S. (Adapted from [7]) 3

1.3 The intersection of aviation research interests and impact of weather 4

1.4 DL1854 flight path visualization 4

1.5 Example of the flight plan generated by SimBrief 5

1.6 FAA Air Route Traffic Control Centers (ARTCCs) 7

1.7 High altitude Instrument Flight Rule (IFR) en-route chart around Atlanta . . 7

2.1 Control Display Unit (CDU) of the Flight Management System (FMS) . . . 14

2.2 Concept of the Dynamic Weather Routes (DWR) 16

2.3 Traffic Aware Planner (TAP) installation in the cockpit 18

2.4 ForeFlight application product 19

2.5 Example routes generated by the PARTNER (Reprinted from [33]) 24

2.6 Flight trajectory comparison between DL1854 and RTOP-v1 25

3.1 Notional diagram of the ANN model structure 28

3.2 Structure of the human brain (Adapted from [37]) 28

3.3 Schematic of the Perceptron 29

3.4 Structure of the MLP 30

3.5 Simple MLP structure as an example 31

3.6 Milestones in the development of the ANN (Reprinted from [43]) 33

3.7 Notional sketch of the SVR 34

3.8 Notional sketch of deploying a non-linear kernel function 35

3.9 Types of data points in the DBSCAN (Adapted from [52]) 38

3.10 Example of the K-NN classification 39

3.11 Example of the K-Dimensional tree decomposition 41

3.12 Notion of consistency in the A* search algorithm 44

3.13 Canonical example of how the A* search algorithm works 45

4.1 Notional sketch of the necessity for continuous wind information 50

4.2 Steps for Research Experiment 1.1 53

4.3 CIWS product visualization generated by the MIT CoSPA system 54

4.4 AWC convective SIGMET polygon at 2019-03-23 23:00 UTC 56

4.5 AA1300 flight path visualization with the AWC convective SIGMET data . 56

4.6 AA1300 flight path in-depth analysis with ground-based weather radar data 58

4.7 Visual comparison of the two pathfinding algorithms (Adapted from [86]) . 61

4.8 DL2143 flight path visualization (en-route only) with the planned waypoints 62

4.9 Decomposition and recomposition of the interpretation of the ASDL Ph.D. process (Adapted from [90]) 65

5.1 Research roadmap 71

5.2 Research overview 72

5.3 GFS wind visualization at 2020-08-20 06:00 UTC (Altitude = 250 hPa) . . 74

5.4 NEXRAD coverage visualization 75

5.5 Selected NEXRAD visualization at 2019-10-06 15:00 UTC 76

5.6 AWC convective SIGMET data visualization at 2019-10-06 15:00 UTC . . 77

5.7 METAR symbol sample (Reprinted from [95]) 78

5.8 Wind-related METAR visualization at 2019-05-18 21:00 and 22:00 UTC . . 79

5.9 AWC PIREP visualization with the raw text data at 2019-10-06 14:50 UTC 80

5.10 Live flight tracking service by FlightAware 81

5.11 FAA pre-determined designated points and ATS routes 82

5.12 Flowchart of the MLP-based wind regression modeling process 83

5.13 Diagram of the MLP-based wind regression model structure 84

5.14 Notional sketch of the hybrid DoE 84

5.15 Loss curve for the final MLP wind regression model 85

5.16 Notional sketch of a grid search approach for SVR hyperparameter selection 86

5.17 IDW method for GFS wind temporal interpolation 88

5.18 Actual vs. Predicted plots of the supervised machine learning-based wind regression models 91

5.19 Validation points for the comparison between linear interpolation and supervised machine learning-based regression 92

5.20 Wind measurement data by the weather balloons at 2021-02-02 12:00 UTC (Altitude = 250 hPa) 93

5.21 GFS wind visualization at 2021-02-02 12:00 UTC (Altitude = 250 hPa) . . 94

5.22 Actual wind measurement vs. Machine learning-based wind prediction . . . 95

5.23 Overview of the unsupervised machine learning-based convective weather polygon generation process 96

5.24 Selected NEXRAD site location visualization 97

5.25 Filtered METAR report visualization 98
5.26 DBSCAN results for the filtered data points 99
5.27 Polygon generation with the clustered data points 99
5.28 NEXRAD and polygon visualization at 2021-02-12 20:00 UTC 101
5.29 Scenario sequence of AA1300 case study 102
5.30 AWC convective SIGMETs vs. New polygons generated by Enabler 2 . . . 103
5.31 U.S. airspace infrastructure (high altitude only) 105
5.32 U.S. airspace infrastructure within the bounding box 106
5.33 Steps for calculating of aircraft ground speed 109
5.34 Notional sketch of how the collision check function works 111
5.35 Notional sketch of the necessity for the Ball Tree implementation 113
5.36 Overview of the additional free-flight route generation process 115
5.37 Notional sketch of the necessity to generate additional free-flight routes . . 116
5.38 Notional process of the proposed methodology 117
5.39 DL971 flight path visualization 118
5.40 Radar intensity of hurricane Laura at 2020-08-27 18:00 UTC 119
5.41 Simulation results generated by the RTOP 121
5.42 Software architecture used for this research 123

6.1 Selected flight datasets used for statistical analysis 127
6.2 Clustering results for the selected historical flights 129
6.3 Median representative flight selection for DFW-PHX flights 130
6.4 Representative flight selection for DFW-DCA flights 131

6.5 Representative flight selection for ORD-IAH flights 132
6.6 Steps for estimating travel time 133
6.7 Apples-to-apples comparison between real flights and simulation cases . . . 134
6.8 DL1944 flight path visualization 135
6.9 DL1944 mission profile 135
6.10 Results of the Hausdorff distance for the representative flight cases 136
6.11 Simulation results for the median cases of the Hausdorff distance box plot . 137
6.12 Simulation results for the outlier cases of the Hausdorff distance box plot . . 139
6.13 Simulation results for the DL1072 flight case 140
6.14 Simulation results for the UAL1872 flight case 141
6.15 Distribution of the potential benefits for all of the representative flights . . . 142
6.16 Simulation results for the UAL389 flight case 142
6.17 O-D great circle distance vs. Elapsed time for representative flight cases . . 144
6.18 Overview of the RTOP-v3 process 145

LIST OF ACRONYMS

AA American Airlines

ADS-B Automatic Dependent Surveillance-Broadcast

AEDT Aviation Environmental Design Tool

AI Artificial Intelligence

ANN Artificial Neural Network

ANSP Air Navigation Service Provider

ARTCC Air Route Traffic Control Center

ASDL Aerospace Systems Design Laboratory

ATC Air Traffic Control

ATL Hartsfield-Jackson Atlanta International Airport

ATS Air Traffic Service

AWC Aviation Weather Center

BFGS Broyden–Fletcher–Goldfarb–Shanno

CCD Central Composite Design

CDU Control Display Unit

CIWS Corridor Integrated Weather System

CoSPA Consolidated Storm Prediction for Aviation

CWAM Convective Weather Avoidance Model

CWAP Convective Weather Avoidance Polygon

DBSCAN Density Based Spatial Clustering of Applications with Noise

DoE Design of Experiment

DTA Design Time Assurance

DWR Dynamic Weather Routes

EASA European Union Aviation Safety Agency

ECMWF European Center for Medium-Range Weather Forecast

EFB Electronic Flight Bag

FAA Federal Aviation Administration

FACET Future Air traffic management Concepts Evaluation Tool

FMS Flight Management System

GA Genetic Algorithm

GFS Global Forecast System

GP Gaussian Process

HDBSCAN Hierarchical Density Based Spatial Clustering of Applications with Noise

HRRR High Resolution Rapid Refresh

ICAO International Civil Aviation Organization

IDW Inverse Distance Weighting

IMA Integrated Modular Avionics

K-NN K-Nearest Neighbor

LAX Los Angeles International Airport

LCC Lambert Conformal Conic

LHS Latin Hypercube Sampling

LSTM Long Short-Term Memory

MCP Multi-Core Processors

MCTS Monte Carlo Tree Search

MERRA-2 Modern-Era Retrospective analysis for Research and Applications-2

METAR Meteorological Aerodrome Report

MFCR Multi-Flight Common Routes

MFE Model Fit Error

MIT Massachusetts Institute of Technology

ML Machine Learning

MLP Multi Layer Perceptron

MRE Model Representation Error

NASA National Aeronautics and Space Administration

NASCENT National Airspace Constraint Evaluation and Notification Tool

NCEP National Centers for Environmental Prediction

NEXRAD Next Generation Radar

NOAA National Oceanic and Atmospheric Administration

ORD Chicago O'Hare International Airport

PARTNER Parametric Real Time Navigation En Route

PIREP Pilot Report

PSO Particle Swarm Optimization

RAP Rapid Refresh

RBF Radial Basis Function

RMSE Root Mean Square Error

RTA Run-Time Assurance

RTOP Real-time Trajectory OPtimization

SA Simulated Annealing

SFO San Francisco International Airport

SID Standard Instrument Departure

SIGMET Significant Meteorological Information

STAR Standard Terminal Area Approach

SVM Support Vector Machine

SVR Support Vector Regression

TAP Traffic Aware Planner

TAS True Airspeed

TDS Text Data Server

TFR Temporary Flight Restriction

TRACON Terminal Radar Approach Control Facilities

U.S. United States

UAM Urban Air Mobility

UTC Universal Time Coordinated

VFR Visual Flight Rules

WHO World Health Organization

SUMMARY

While the Coronavirus disease of 2019 (COVID-19) pandemic has greatly impacted the number of flights and passengers, flight delay-related challenges may reappear soon as aviation traffic continues to rebound from the pandemic. Since flight delays are primarily caused by weather, airlines typically gather all available weather information before departure to generate flight routes that avoid hazardous weather while minimizing operating expenditures. However, pilots potentially have to perform in-flight re-planning as weather information can significantly change after original flight plans are created.

Current in-flight re-planning systems rarely cause accidents in U.S. airspace; however, one potential issue is that the systems are not fully automated; thus, pilots today perform some portions of the in-flight activities manually. The manual decision-making process may not be an issue at this moment; however, the advent of new communication systems (e.g., on-board fast connectivity technology) will bring more information into the cockpit. This has the potential to significantly increase the workload of pilots especially if they must consider a large volume of information (i.e., information overload), leading to potential safety issues in the near future. Another potential issue is that weather forecasts used for current in-flight re-planning systems are not always accessible in a timely manner.

This research attempts to resolve the aforementioned potential issues by developing a framework that automatically performs in-flight re-planning continuously with the latest weather information sets available. This study specifically develops 1) a supervised machine learning-based wind prediction model to obtain continuous wind information, 2) an unsupervised machine learning-based short-term (i.e., every 10 minutes) convective weather prediction model to define reliable and up-to-date areas of convective weather, and 3) a designated points-based flight path optimization model that combines the A* search algorithm with a free-flight approach to find an optimal flight path. The intent of this research is to provide an automated framework with two use-cases in mind: 1) to help flight

dispatchers at major airlines of the U.S. by providing the capability to continuously re-route flights using the latest weather information and 2) to alleviate the cockpit workload of pilots employed by small airline operators (e.g., private business jets) that do not necessarily have flight dispatchers but rather ask the pilots to generate and update new flight plans.

As a part of this research, statistical analyses are performed using real flights (e.g., American Airlines, Delta Airlines, and United Airlines) to prove the potential benefits and applicability of the proposed methodology. The results indicate that the framework developed by this research generates flight routes that reduce flight time by up to two percent in most cases. The outcome of this research establishes not only an automated framework that enables the airlines to continuously perform flight path optimization in the contiguous U.S. more accurately and frequently but also provides a basis for optimizing flight routes for all categories of airplanes such as private business jets.

CHAPTER 1
INTRODUCTION

1.1 Research Motivation

Coronavirus disease of 2019 (COVID-19) has led the World Health Organization (WHO) to declare a global pandemic in January 2020 [1]. The pandemic has greatly impacted all industries around the globe by creating significant uncertainties. The aviation industry is no exception to this worldwide trend. According to reports by the International Civil Aviation Organization (ICAO) in 2020, there is an approximately $390 billion operating loss of commercial airlines compared to 2019 [2].

Despite increasing concerns surrounding the COVID-19 crisis, worldwide airline traffic has made a slow and steady rebound from the pandemic for the following reasons: First, various aviation industry committees have established aviation public health procedures such as COVID-19 aviation health safety protocol by the European Union Aviation Safety Agency (EASA) [3]. Second, airlines have implemented a variety of procedures to mitigate the spread of COVID-19 by installing thermal cameras to potentially identify infected passengers. Third, airports have enhanced self-service options to eliminate face-to-face virus transmission to airport personnel by implementing interface-free check-in and fully autonomous security procedures. In addition to these efforts by the airlines, full recovery from the pandemic is expected with the development and distribution of effective COVID-19 vaccines.

While the pandemic has greatly impacted the number of flights and passengers, the fact that long-term forecasts for aviation predict significant growth in the coming years implies that traffic-related challenges may reappear soon [4]. As aviation traffic rebounds as shown in Figure 1.1 [5], many airlines are concerned about flight delays because such

delays are directly linked to higher fuel burn, increased labor expenditures, and an overall rise in aircraft operating costs. For example, the total cost of delays for United States (U.S.) airlines in 2018 amounted to $28.2 billion as shown in Table 1.1 [6]. There are many factors that affect flight delays. Some factors are under the direct control of airlines such as aircraft turnarounds between flights; however, there are perhaps even more reasons that are outside of the airline's control such as weather. In fact, most delays are caused by convective weather (e.g., thunderstorms) and these delays tend to peak during the summer months as shown in Figure 1.2 [7].

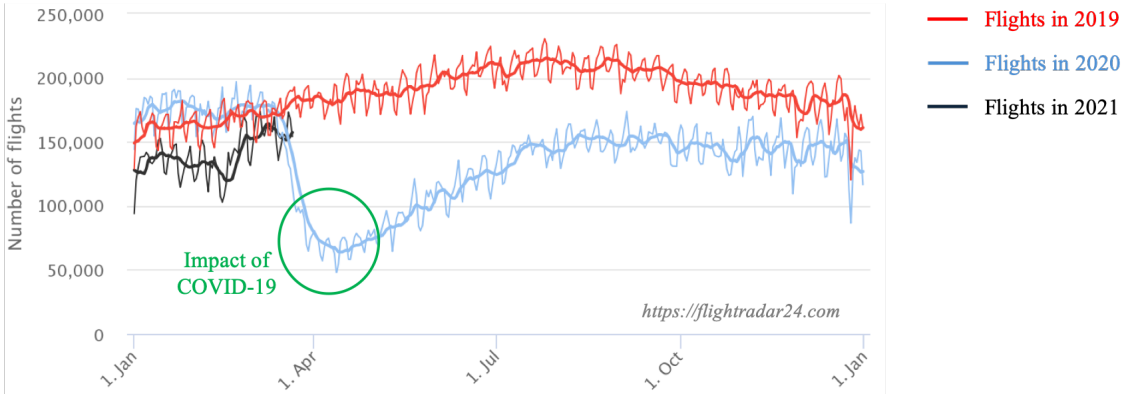


Figure 1.1: Total number of flights tracked by Flightradar24 (Adapted from [5])

Table 1.1: Total cost of delay (unit: $billion)

	2012	2013	2014	2015	2016	2017	2018
Indirect	2.5	2.7	2.6	3.1	3.0	3.4	3.6
Airlines	5.7	6.0	5.8	5.8	5.6	6.4	6.4
Passengers	9.7	11.0	10.5	13.3	13.3	14.8	16.1
Lost demand	1.3	1.4	1.4	1.8	1.8	2.0	2.1
Total	19.2	21.1	20.3	24.0	23.7	26.6	28.2

With the increase in aviation congestion, airlines are also concerned about fuel expenses given that fuel accounts for up to 35 percent of airline operating costs [8]. Airlines have a

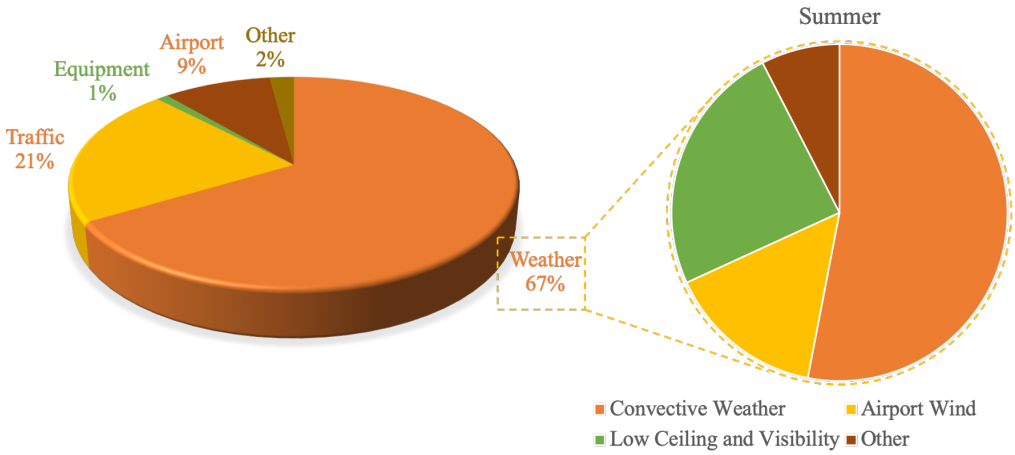


Figure 1.2: Sources of flight delays in the U.S. (Adapted from [7])

variety of options to reduce fuel consumption. For example, airlines can buy new aircraft and fuel-efficient engines; however, the required investment to purchase them may be expensive for the airlines. As an operational solution, airlines can employ less costly options to optimize fuel consumption. One example is that airlines most commonly require pilots to optimize their day-to-day operations by taking advantage of favorable winds (i.e., riding tailwinds but avoiding headwinds).

For instance, one Virgin Atlantic flight from Los Angeles to London on February 19th, 2019 achieved record-breaking speeds over central Pennsylvania due to significant tailwinds (i.e., the ordinary cruising speed of such a flight is approximately 561 mph but the Virgin Atlantic flight peaked at a whopping 801 mph due to tailwinds) such that the flight arrived 48 minutes early even though the flight did not remain in the jet stream for long [9]. Given the aforementioned observations, it is undeniable that weather is a primary concern for airlines as it affects fuel consumption and the potential for flight delays as illustrated in Figure 1.3.

To take into account weather-related flight operation concerns, airlines typically initiate day-to-day flight planning activities for safe and efficient flights from takeoff to landing. While airlines have different flight planning strategies that depend on policies, procedures,

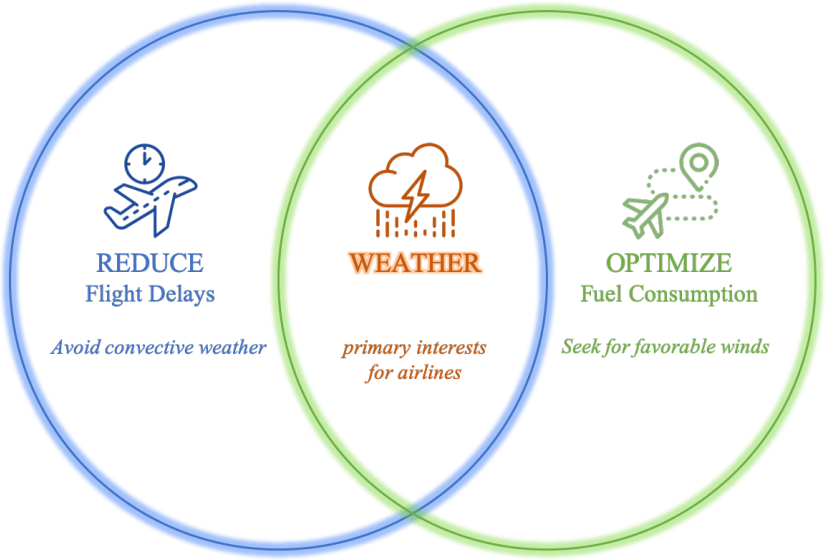


Figure 1.3: The intersection of aviation research interests and impact of weather

and aircraft capabilities, the most common flight planning strategy in the U.S. is to guide aircraft away from areas of convective weather, especially during the summer season.

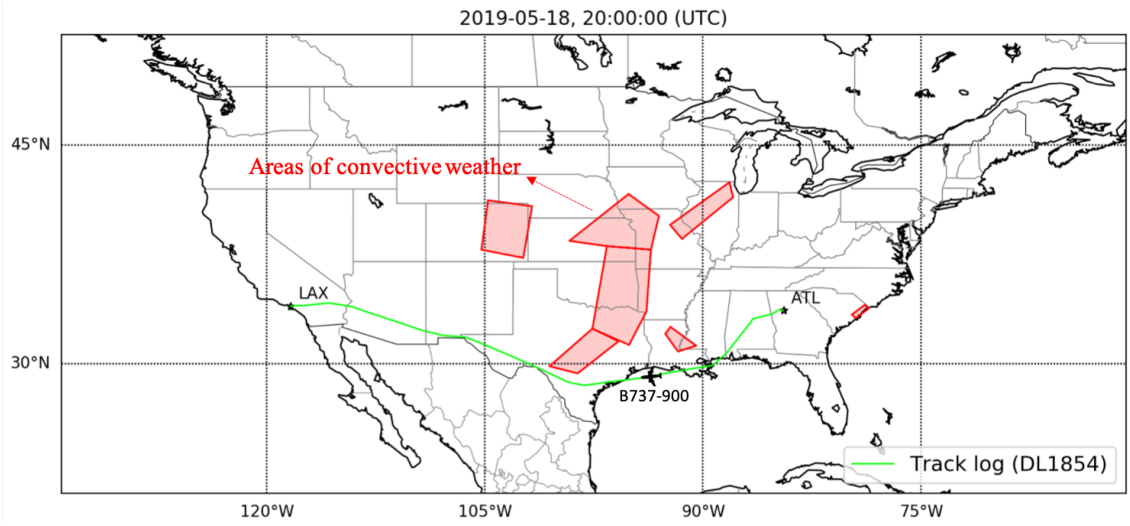
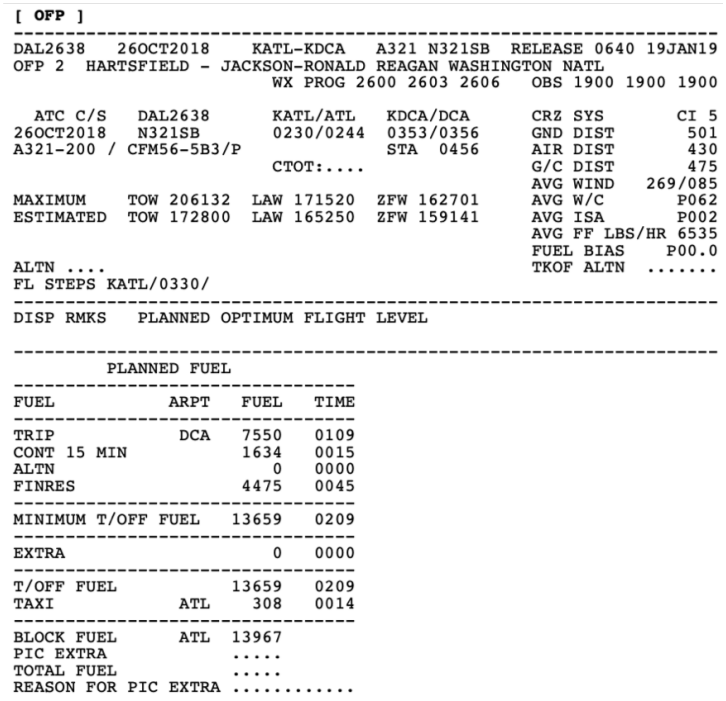


Figure 1.4: DL1854 flight path visualization

For example, Figure 1.4 shows the trajectory of Delta Airlines 1854 Flight from Hartsfield-Jackson Atlanta International Airport (ATL) to Los Angeles International Airport (LAX)

on May 18th, 2019. As can be seen, the flight avoided areas of convective weather activity, resulting in a circuitous routing taking 5 hours and 31 minutes. Given that a flight from ATL to LAX generally takes 4 hours and 30 minutes under clear weather conditions, this indicates that convective weather can lengthen flights, resulting in additional expenditures for airlines.

For this reason, airlines typically gather all available weather information before departure to generate flight routes that avoid hazardous weather while also minimizing operating expenditures. The airlines hire highly trained flight dispatchers to coordinate these pre-flight activities that include ensuring up-to-date paperwork, checking the weather, generating flight plans, and coordinating with various authorities in order to ensure safe, legal, and cost-efficient flights. The flight dispatchers are mainly responsible for 1) estimating fuel consumption, 2) seeking favorable winds to optimize flight operations, and 3) using the latest weather forecasts to guide aircraft away from areas of convective weather [10].



```
[ OFP ]
------------------------------------------------------------------------
DAL2638   26OCT2018    KATL-KDCA   A321 N321SB  RELEASE 0640 19JAN19
OFP 2  HARTSFIELD - JACKSON-RONALD REAGAN WASHINGTON NATL
                        WX PROG 2600 2603 2606   OBS 1900 1900 1900

  ATC C/S   DAL2638      KATL/ATL   KDCA/DCA      CRZ SYS       CI 5
26OCT2018   N321SB       0230/0244  0353/0356     GND DIST       501
A321-200 / CFM56-5B3/P               STA  0456     AIR DIST       430
                        CTOT:....                 G/C DIST       475
                                                  AVG WIND   269/085
MAXIMUM    TOW 206132  LAW 171520  ZFW 162701     AVG W/C       P062
ESTIMATED  TOW 172800  LAW 165250  ZFW 159141     AVG ISA       P002
                                                  AVG FF LBS/HR 6535
                                                  FUEL BIAS    P00.0
ALTN ....                                         TKOF ALTN  .......
FL STEPS KATL/0330/
------------------------------------------------------------------------
DISP RMKS   PLANNED OPTIMUM FLIGHT LEVEL

------------------------------------------------------------------------
         PLANNED FUEL
---------------------------------
FUEL           ARPT   FUEL   TIME
---------------------------------
TRIP            DCA   7550   0109
CONT 15 MIN           1634   0015
ALTN                     0   0000
FINRES                4475   0045
---------------------------------
MINIMUM T/OFF FUEL   13659   0209
---------------------------------
EXTRA                    0   0000
---------------------------------
T/OFF FUEL           13659   0209
TAXI            ATL    308   0014
---------------------------------
BLOCK FUEL      ATL  13967
PIC EXTRA            .....
TOTAL FUEL           .....
REASON FOR PIC EXTRA .............
```

Figure 1.5: Example of the flight plan generated by SimBrief

For example, Figure 1.5 [11] provides an example of the flight plan document generated by a flight dispatcher system called SimBrief. As can be seen, the document includes a variety of details such as flight number (i.e., DAL2638), flight date (i.e., 26OCT2018), aircraft type (i.e., A321), and departure and arrival airport identifiers (i.e., KATL-KDCA). This document also provides the section called Planned Fuel, which offers a breakdown of the required fuel load.

Once an initial flight plan is created by a flight dispatcher, a pilot activates the flight plan with Air Traffic Control (ATC) prior to departure. The pilot stays in contact with ATC during taxi and departure. After the departure procedure, the pilot begins to follow the flight path generated by the flight dispatcher. It is important to note that changes do not typically occur during the departure phase as the procedures are strictly standardized to ensure every aircraft maintains a safe flight during takeoff. During the en-route phase, however, the pilot may have to request changes (i.e., performing in-flight re-planning) because weather information (e.g., areas of convective weather) can significantly change after the original flight plan is created. When performing in-flight re-planning, the pilot generally communicates with 21 Air Route Traffic Control Center (ARTCC) in the U.S. and requests a change if needed due to a hazardous weather condition. Each center supervises thousands of square miles encompassing several states within controlled U.S. airspace as shown in Figure 1.6 [12] and hands over its responsibility when aircraft borders another center's area.

Although the en-route phase is not heavily constrained by a procedure, pilots are still guided to follow waypoints that are pre-determined with longitude and latitude coordinates in U.S. airspace (i.e., highways in the sky) as shown in Figure 1.7 [13]. However, it is important to note that pilots do not always follow the waypoints, especially when they encounter a hazardous weather condition. In case there is a need to change flight routes (i.e., in-flight re-planning) due to hazardous weather activity, pilots typically ask the ARTCC to change it in a lateral direction as air traffic tends to flow in different directions based on altitude (e.g., air traffic flows east in 36,000 feet and moves west in 35,000 feet).

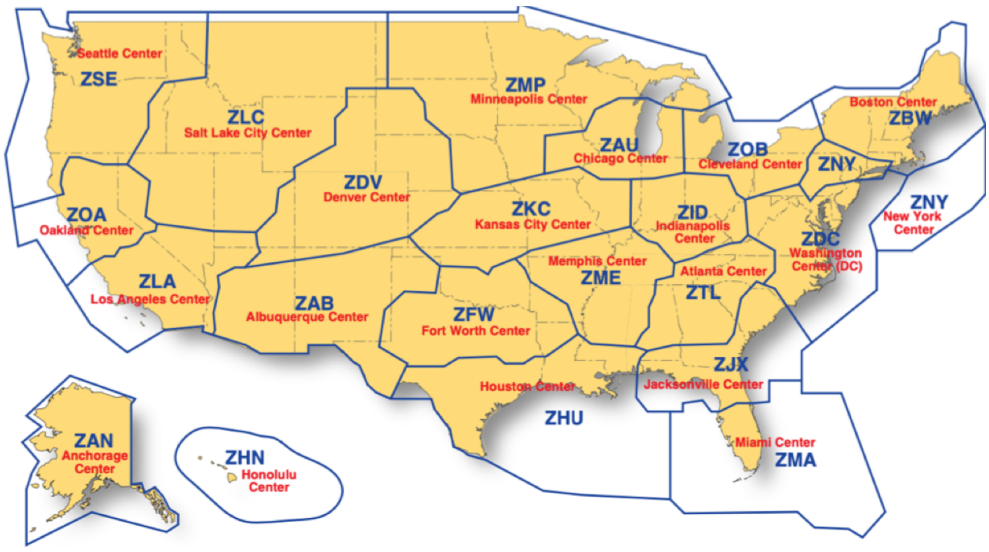


Figure 1.6: FAA Air Route Traffic Control Centers (ARTCCs)

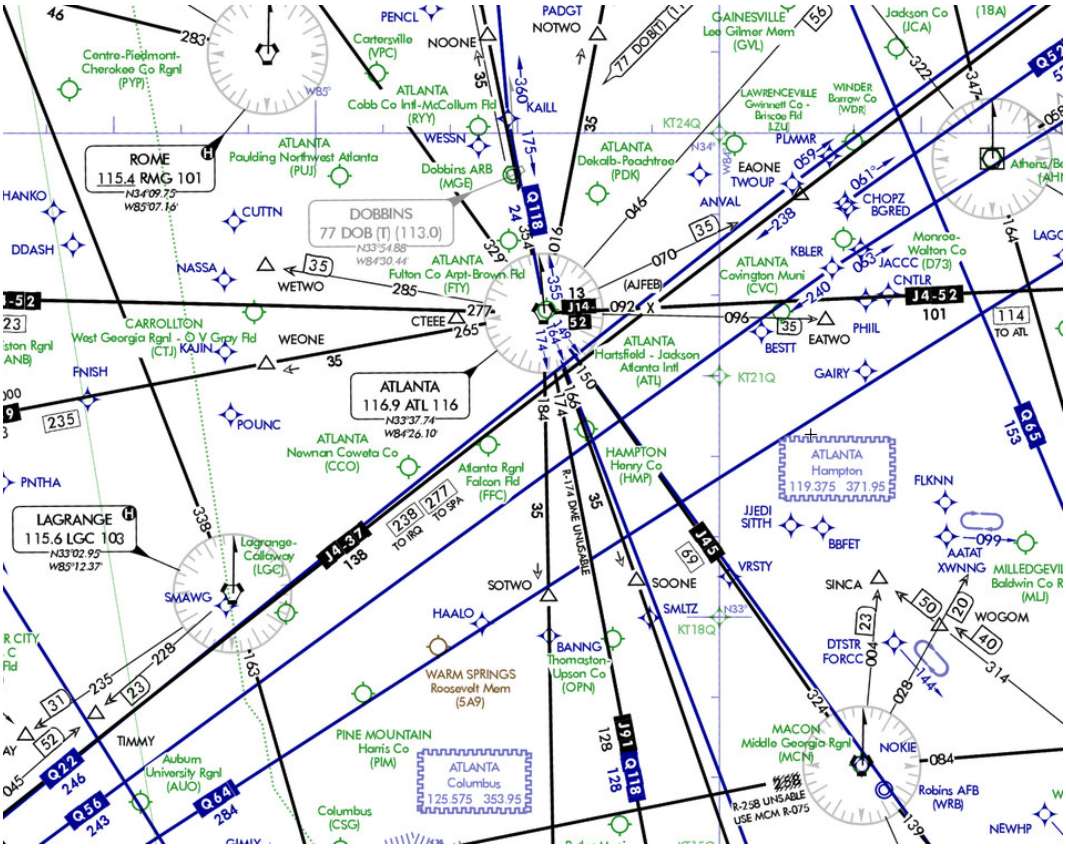


Figure 1.7: High altitude Instrument Flight Rule (IFR) en-route chart around Atlanta

While it appears that current in-flight re-planning systems result in very few accidents in U.S. airspace, there are still a few potential issues identified from the interview with the pilots in this research. Appendix A provides the transcript of the pilot interviews.

Potential Issue 1: Current in-flight re-planning systems are not fully automated; thus, pilots today perform some portions of the in-flight activities manually.

The manual decision-making process may not be an issue at this moment; however, the advent of higher bandwidth communication systems will soon bring more information into the flight deck enabling flight crews to get the latest information about the weather and enabling them to perform continuous re-routing in a more frequent manner. Therefore, this has the potential to significantly increase the workload of flight crews especially if they must consider a large volume of information (i.e., information overload), leading to potential safety issues in the near future.

Potential Issue 2: Weather forecasts used for current in-flight re-planning systems are not always accessible in a timely manner.

It is critical to obtain reliable weather information in a timely manner for performing in-flight re-planning. Another potential issue is that most weather forecasts used for current in-flight re-planning systems use relatively sparse (i.e., discrete) information and may not always be accessible in real-time; thus, it challenges pilots to perform in-flight re-planning more accurately and frequently.

1.2 Research Objective

This research attempts to resolve the aforementioned potential issues by developing an automated framework that performs in-flight re-planning continuously with the latest weather information sets available. In summary, the objective of this research is as follows:

Research Objective: Develop an automated framework that performs in-flight re-planning continuously with the latest weather information sets available

The intent of this research is to provide the automated framework with two use-cases in mind: 1) to help flight dispatchers at major airlines of the U.S. by providing the capability to continuously re-route flights using the latest weather information sets available and 2) to alleviate the cockpit workload of pilots employed by small airline operators that do not necessarily have flight dispatchers but rather ask pilots to generate and update flight plans. More specifically, given that the pilots employed by the small airline operators manually deal with a variety of in-flight activities (e.g., the pilots enter flight plan information manually into the system using a keyboard or touchscreen), it is expected that the automated framework reduces the workload of the pilots by automatically generating and updating new flight plans with relevant information such as estimated arrival time and fuel consumption.

To achieve these objectives, this research uses a supervised machine learning algorithm to yield a continuous wind prediction model, utilizes an unsupervised machine learning algorithm to forecast reliable and up-to-date convective weather activity, and proposes a designated points-based flight path optimization model to optimize flight routes. In this way, the framework developed by this research can mitigate weather-related flight delays and associated operating expenditures by providing novel flight path optimization capabilities that use the latest weather information. Additionally, the framework is intended to be first used by commercial airlines and business aviation but ultimately by all other aviation domains such as Urban Air Mobility (UAM).

1.3 Research Scope

This research specifically concentrates on the en-route phase of flight for the following reasons: First, the en-route phase of flight is typically the dominant segment of flights,

especially if long-haul flights are considered. Second, the en-route phase of flight is not heavily constrained compared to the other flight phases such as departure and arrival that are generally standardized to ensure flight safety. This implies that the en-route phase of flight potentially has more opportunities for improvements (e.g., flight path optimization) than the other flight phases. The scope of this research is also limited to the U.S. territory only. Therefore, this research focuses on a flight path optimization problem that pertains only to en-route flights in the contiguous 48 states of the U.S. with the aim of minimizing weather-related delays and optimizing fuel efficiency.

This research also relies on the following assumptions in the computer simulation environment: First, aircraft True Airspeed (TAS) is constant during the en-route phase of flight. Second, convective weather activity is represented by polygons associated with high penalty costs if the route penetrates the polygons, meaning that the algorithm must avoid the polygons to optimize flight routes. Third, aircraft basically follow established waypoint-to-waypoint flight routes only except in situations where hazardous weather is encountered. Last, Temporary Flight Restriction (TFR) and traffic-related issues are not considered, indicating that the focus of the evaluation is on a single aircraft during in-flight phases.

It is important to note that the framework developed by this research has to be equipped with a broadband aviation connectivity framework such as the SmartSky 4G LTE product [14] in order to enable timely connections to relevant data aggregated from multiple sources, meaning that there is an assumption that no connection issues exist during in-flight.

1.4 Dissertation Structure

Chapter 1 introduced the research motivation, research objective, and research scope. In summary, this research is initiated to attempt to resolve potential issues of the current in-flight re-planning systems (e.g., manual decision-making). The objective of this research

is to develop a framework that automatically performs in-flight re-planning continuously with the latest weather information sets available. The remainder of this dissertation is organized as follows.

Chapter 2 aims to review previous work related to the research objective presented in this dissertation, which helps identify research gaps. Chapter 3 provides relevant background information on machine learning and optimization techniques that are used in this research. This chapter may be regarded as optional for readers who have little knowledge of the techniques. Chapter 4 presents the formulation of research problems with a series of research questions, research hypotheses, and research experiments.

Chapter 5, which is the main focus of this research, illustrates the implementation of the proposed methodology. Chapter 6 constructs a set of simulation scenarios and presents a series of results obtained through statistical analyses. Finally, Chapter 7 concludes this dissertation by providing the recapitulation of research statements, by summarizing the contributions of this research, by recommending for future work to this line of research, and outlining potential applications.

CHAPTER 2
LITERATURE REVIEW

In relation to the research objective mentioned from section 1.2, several attempts have been made over the years to improve the continuous flight re-routing process. This chapter aims to review previous work related to the research objective, which helps identify research gaps.

2.1 Flight Management System

A Flight Management System (FMS) is a specialized computer system implemented into the cockpit designed to reduce the workload of the flight crew by providing an automated system dealing with a variety of in-flight activities. The FMS typically comprises two main components (i.e., flight management computer and display unit) and provides the following functionalities [15]: navigation function, flight planning function, trajectory prediction function, performance function, and guidance function.

One of the main functionalities of the FMS is to manage a flight plan. A flight plan is typically determined on the ground before departure either by a pilot for small airline operators or a highly trained flight dispatcher for major airlines. However, pilots potentially have to change the original flight plan for various reasons (e.g., encounter hazardous weather conditions) and they use the FMS to modify the flight plan if needed by linking data stored in the navigation database such as: 1) Standard Instrument Departure (SID) procedures, 2) Standard Terminal Area Approach (STAR) procedures, 3) airways, 4) en-route waypoints, and 5) pre-stored company routes. The flight plan modification normally comes either from the pilot's selections or the communications with flight dispatchers (e.g., flight dispatchers recommend company routes or specific routes for the particular situation). The modified version of the flight plan is then transmitted to the trajectory prediction functionality to

compute the predicted aircraft profile with the new flight plan. This process is periodically repeated and updated as needed.

While it is true that the FMS reduces the workload of the flight crew by implementing the automation system, it is important to note that the system still has some portions that are not fully automatic but manual input. For example, when pilots need to modify the initial flight plan as the pilots receive the change request from flight dispatchers, the pilots generally enter information manually by using either a keyboard or touchscreen. Figure 2.1 [16] shows an example of the Control Display Unit (CDU) of the FMS where the pilot is supposed to enter flight plan input manually. The flight plan used in this example originates at LAX and terminates at San Francisco International Airport (SFO) as shown in Figure 2.1a. Also, Figure 2.1b shows the scenario where the pilot deletes the waypoint LAX from the flight plan. Based on the aforementioned observations, the following research gap can be identified:

Research Gap 1: A Flight Management System (FMS) reduces the workload of the flight crew by automating a variety of in-flight activities but some parts in the flight planning functionality are still manual input.

2.2 Ground-based Flight Planning Framework

Flight planning mainly considers aircraft performance, air traffic, navigation procedures, and weather to answer the following aspects [17]: 1) how much fuel is required and 2) what does the route look like. Since it is a complex multi-disciplinary process, major airlines typically hire flight dispatchers to deal with the process. Flight dispatchers are responsible for coordinating all the preparations required to ensure a safe and efficient flight. Prior to departure, they typically examine multiple information sources and integrate the sources to create an initial flight plan. More specifically, the pre-flight planning process is illustrated as follows [18]: 1) enter all necessary information (e.g., origin and destination) into a

(a) Invoking departure and arrival airports

(b) Deleting waypoints from the flight plan

Figure 2.1: Control Display Unit (CDU) of the Flight Management System (FMS)

system, 2) find a recommended route from a company, 3) see if the recommended route penetrates areas of convective weather, 4) try to pick another route recommended from the company if the original route penetrates any areas, 5) create a new route based on multiple information sources if none of the company routes are available, and 6) release the flight plan.

While the pre-flight planning process is straightforward during clear weather conditions, flight dispatchers are prone to high mental workload during severe weather conditions as they need to manually re-route many flights, potentially leading to yield inefficient flight plans. To support these manual procedures, National Aeronautics and Space Administration (NASA) has developed and tested a ground-based flight planning framework over the past several years. The framework is named the Future Air traffic management Concepts Evaluation Tool (FACET) developed by NASA Ames Research Center [19].

Since the FACET was mainly developed to detect aircraft conflict, assess airspace congestion, and estimate air traffic controller workload within a simulation, the weather avoidance capability within the FACET was only implemented later using the concept of the Dynamic Weather Routes (DWR) developed by NASA. Routing updates proposed by the DWR are triggered whenever a new routing opportunity saving 5 minutes or more of flight time arises; then auxiliary waypoints are added as needed to avoid convective weather along the initial flight route as shown in Figure 2.2 [20]. In addition to the DWR, NASA also developed the Multi-Flight Common Routes (MFCR) concept. The National Airspace Constraint Evaluation and Notification Tool (NASCENT) is an application that implements the DWR and MFCR concepts into the FACET environment.

Although ground-based flight planning tools have mostly automated the task of flight planning for flight dispatchers, pilots may have been suffered from different problems such as a lack of weather information during flight. For example, when pilots encounter an unanticipated weather condition, they generally request a change to the ARTCC in terms of the flight route; however, pilots today are occasionally not permitted to make changes for the

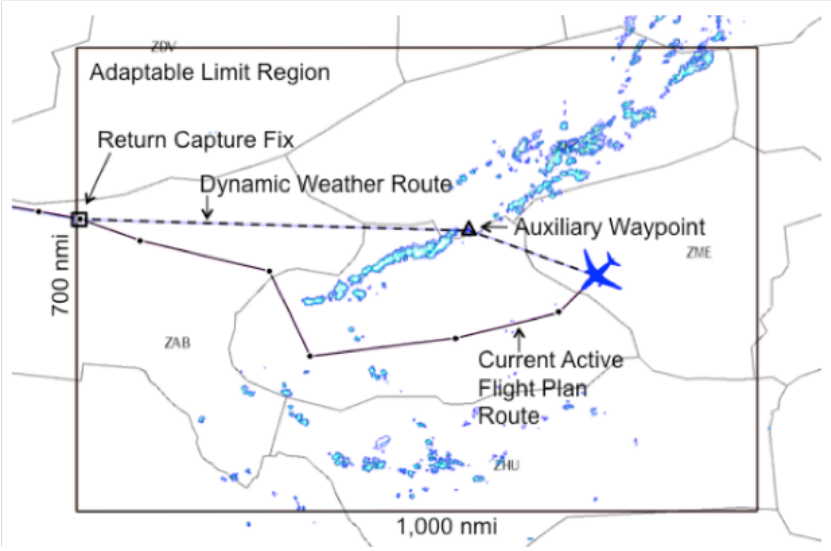


Figure 2.2: Concept of the Dynamic Weather Routes (DWR)

following reasons: First, the request may be denied by the ARTCC because pilots do not have the same knowledge as the ARTCC. Second, the request may subsequently be deferred by the ARTCC because pilots today are not typically allowed to monitor for optimization opportunities with weather information that is continuously updated. Such disapproved requests can be inefficient to everyone involved in the process (e.g., pilots, flight dispatchers, and controllers) and cost workload for them. Based on the aforementioned observations, the following research gap can be identified:

> *Research Gap 2: A ground-based flight planning framework has mostly automated the task of flight planning for flight dispatchers but the framework does not address the connectivity issue of co-constructing knowledge between pilots and flight dispatchers, leading to disapproved requests resulting in inefficient communication.*

2.3 Electronic Flight Bag

Pilots traditionally accomplished a variety of in-flight management tasks by using paper-based reference materials such as flight operating manual, navigational charts, and aircraft

operating manual. They literally had to carry a heavy document bag to the cockpit to perform in-flight management duties such as weight and balance calculations. The concept of Electronic Flight Bag (EFB) has been recently introduced with the advent of tablet computing devices such as iPad. The EFB is now considered as the replacement of the traditional paper-based materials in a way that the EFB helps pilots conduct in-flight activities in a more easy and efficient way by providing various digital documents for the in-flight operations. In addition to the concept of EFB, with the introduction of Automatic Dependent Surveillance-Broadcast (ADS-B), a variety of information services are becoming available on the flight deck. In fact, the advent of ADS-B has transformed many segments of civil aviation. For example, an airborne weather radar system became an essential tool to maintain safe and efficient flights.

In this respect, given that a broadband aviation connectivity framework enables timely connections to relevant data from multiple sources, the EFB has been receiving increased attention because it has the potential to automatically perform many complex in-flight tasks that usually carried out by pilots manually. Along with this rapidly changing technology, many aviation-related applications (e.g., in-flight decision support tool) have been developed to be fed into the EFB.

Traffic Aware Planner (TAP)

NASA has developed a cockpit-based flight planning tool as shown in Figure 2.3 [21], named Traffic Aware Planner (TAP), to minimize unnecessary workload between pilots and controllers by increasing the likelihood of approval of change requests. The TAP was evaluated by the pilots in the eastern and north-eastern U.S. airspace. As a result, based on the survey, most pilots provided feedback mentioning that the tool would be helpful for them [22]. However, it is important to note that the TAP is designed to generate re-route plans with only 90 minutes look-ahead [23], which potentially leading to be trapped in a local minimum especially for a long-haul flight case.

Figure 2.3: Traffic Aware Planner (TAP) installation in the cockpit

GoDirect

The GoDirect application is a flight planning application developed by Honeywell Aerospace where a worldwide flight planning service is available via phone, mobile devices, or data link communications. Honeywell Aerospace provides the service where the Honeywell's professional flight data specialists assist customers in preparing the most efficient flight plan 24 hours / 7 days a week. More specifically, the application provides the following services: 1) Flight Preview allows pilots to pre-fly instrument approach procedures by viewing the approach from a cockpit perspective as flown over a terrestrial map and 2) The Weather Information allows pilots to view forecast weather for a particular flight plan. It seems that the application primarily focuses on pre-departure flight planning services useful for pilots but the service is still available while aircraft is in the en-route phase. However, it is important to note that the application currently does not have a consistent way to provide a flight planning support service given that each specialist may be recommending a flight plan to customers in their own way (i.e., biased assessment) [24]. More importantly, the application may possibly limit the speed of service if a large volume of flight planning is requested by customers during significant weather events.

ForeFlight

The ForeFlight application is specifically designed for pilots in order to provide airport

information, surface condition, and updated weather information. Figure 2.4 [25] shows an example product of the ForeFlight application implemented in a tablet. The ForeFlight remains a top-selling iPad-based EFB application in the business aviation market because of two main reasons: First, it generally provides accurate flight time and fuel burn numbers. Second, the ForeFlight's graphical weather briefing tool walks users through a detailed briefing in an easily interpreted format. However, it is important to note that the application does not update information in real-time. There is a large delay in the generation of new information and the route does not change based on the new information.

Figure 2.4: ForeFlight application product

Aerobahn

The Aerobahn application is a tool designed to improve pre-departure activities in the airport environment, which is currently being used by the Terminal Radar Approach Control Facilities (TRACON) [26]. While the Aerobahn application has already proved its capability (e.g., the application is able to optimize gate operations), it is important to note that

the application does not give as much weather information as is needed. More importantly, the application primarily focuses on departure and arrival sequences rather than en-route sequences. Based on the aforementioned literature search related to EFB, the following research gap can be identified:

> *Research Gap 3: The most widely used aviation-related applications implemented in the Electronic Flight Bag (EFB) have proved their capabilities in certain application areas; however, the applications may not be feasible for a real-time flight path optimization framework that uses the latest weather information to continuously update flight routes.*

2.4 Machine Learning-based Flight Route Prediction

With the advent of Artificial Intelligence (AI), the research paradigm in various engineering areas has been recently transitioned from theory-based to data-driven approaches. The aviation industry is no exception to this paradigm shift. Machine Learning (ML) techniques are coming with a fast face and being adopted widely in the aviation domain. Regarding this paradigm shift, the EASA claims that AI is still not widely exploited in the aviation industry; thus, the organization prospects the following areas where AI can support complex problems especially in air traffic management systems [27]: 1) improving strategic planning by using visual analytics and ML techniques, 2) enhancing trajectory prediction based on a ML-based aircraft performance model, 3) enabling higher automation in ATC, 4) better understanding passenger behaviour by using the big data skills, 5) increasing the operational efficiency of air traffic control by using several ML-based recognition algorithms, and 6) refining time and wake separation by using time-series ML techniques.

In particular, given that flight re-route negotiation between ARTCCs and pilots is considered as a key component of future air traffic management, many research groups have made an effort on predicting operationally acceptable flight routes using machine learn-

ing techniques. For example, A. D. Evans and et al. [23, 28] proposed an approach that used various supervised machine learning algorithms (e.g., Logistic Regression, AdaBoost, and Random Forest) and the hierarchical clustering technique to dynamically generate operationally acceptable strategic re-route options. The approach was demonstrated for a historical pre-departure flight such as from Dallas/Fort Worth International Airport to Newark Liberty International Airport. Heather Arneson et al. [29] presented a clustering-based binary classification method that quantified the impact of convective weather on pre-departure flights by extracting relevant features from the large volume of weather data available. Yutian Pang et al. [30] used the web-mining tools and Long Short-Term Memory (LSTM) neural network to address the issue of convective weather-related aircraft trajectory prediction prior to departure using the last on-file flight plan and convective weather information.

In addition to predicting operationally acceptable flight routes, it is important for future air traffic management to predict en-route flight time under severe convective weather conditions as the predictions could help flight operators estimate potential flight delays. In response to this concern, many research groups have been committed to developing machine learning-based models that predict estimated en-route flight time and flight delays. For example, Guodong Zhu et al. [31] proposed a method that utilized a variety of machine learning algorithms (e.g., GLM Lasso, XGBoost, and K-NN) with publicly available weather and traffic data to predict en-route flight time before departure. Youngjin Kim et al. [32] used the LSTM deep learning architecture to investigate the effectiveness of machine learning models in the air traffic delay prediction tasks.

The deep learning-based approaches could help flight operators not only predict operationally acceptable flight re-routes using historical flights but also estimate en-route flight time and delays; however, it is important to note that the approaches do not take into account optimizing flight trajectories based on real-time operational conditions (e.g., wind aloft) to produce efficient routes under severe weather. This is perhaps related to a compu-

tational cost issue given that deep learning techniques typically require a massive training time. Based on the aforementioned literature search, the following research gap can be identified:

> *Research Gap 4: Deep learning techniques are prevalently adopted in the aviation industry; however, these techniques may not be feasible for a real-time flight path optimization framework due to high computational costs.*

2.5 Research Gap

In chapter 2, a broad review of the literature was performed under four primary areas: 1) flight management system, 2) ground-based flight planning framework, 3) electronic flight bag, and 4) machine learning-based flight route prediction. The literature search resulted in the following research gaps:

- Research Gap 1: A Flight Management System (FMS) reduces the workload of the flight crew by automating a variety of in-flight activities but some parts in the flight planning functionality are still manual input.
- Research Gap 2: A ground-based flight planning framework has mostly automated the task of flight planning for flight dispatchers but the framework does not address the connectivity issue of co-constructing knowledge between pilots and flight dispatchers, leading to disapproved requests resulting in inefficient communication.
- Research Gap 3: The most widely used aviation-related applications implemented in the Electronic Flight Bag (EFB) have proved their capabilities in certain application areas; however, the applications may not be feasible for a real-time flight path optimization framework that uses the latest weather information to continuously update flight routes.
- Research Gap 4: Deep learning techniques are prevalently adopted in the aviation

industry; however, these techniques may not be feasible for a real-time flight path optimization framework due to high computational costs.

Based on the aforementioned research gaps identified from the literature search, this research established the need to develop a framework that automatically performs in-flight re-planning more accurately and frequently. This naturally led to define an overarching research question as follows:

Overarching Research Question: How can a cockpit-based real-time flight path optimization framework, which automatically performs in-flight re-planning continuously with the latest weather information, be developed?

The Aerospace Systems Design Laboratory (ASDL) at the Georgia Institute of Technology has made a significant effort to answer the overarching research question. Several ASDL-initiated flight path optimization tools are summarized in Table 2.1. In 2018, the ASDL research group partnered with Honeywell Aerospace to develop a tool named the Parametric Real Time Navigation En Route (PARTNER) that optimizes flight routes using the kinematic A* algorithm [33]. Figure 2.5 shows an example route generated by the PARTNER. The red rectangular box represents an area that must be avoided by aircraft.

Table 2.1: ASDL-initiated flight path optimization software

Tool	PARTNER	RTOP-v1	RTOP-v2	RTOP-v3
Path optimization modeling	Yes	Yes	Yes	Yes
Wind modeling	No	Yes	Yes	Yes
Convective weather modeling	No	No	Yes	Yes
Free-flight modeling	No	No	No	Yes

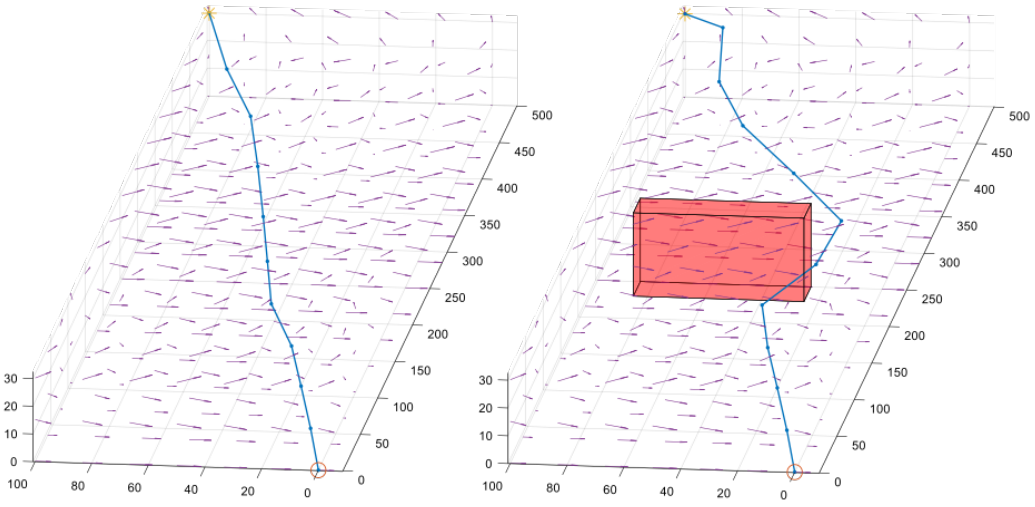
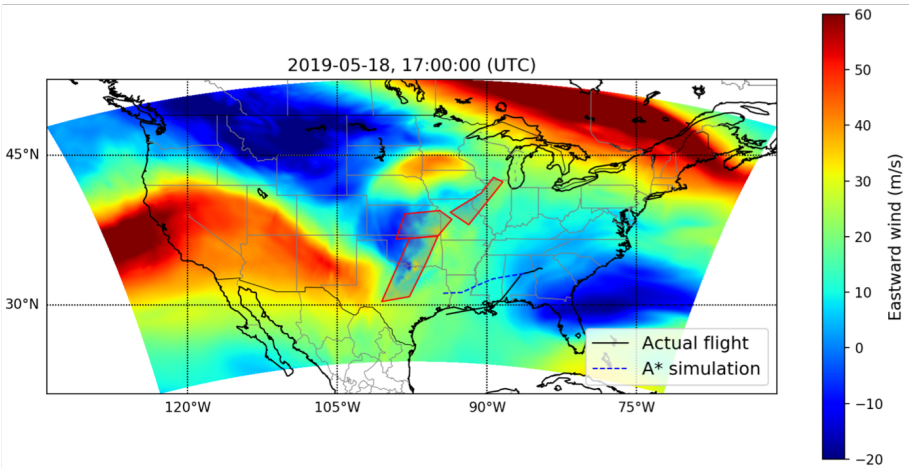


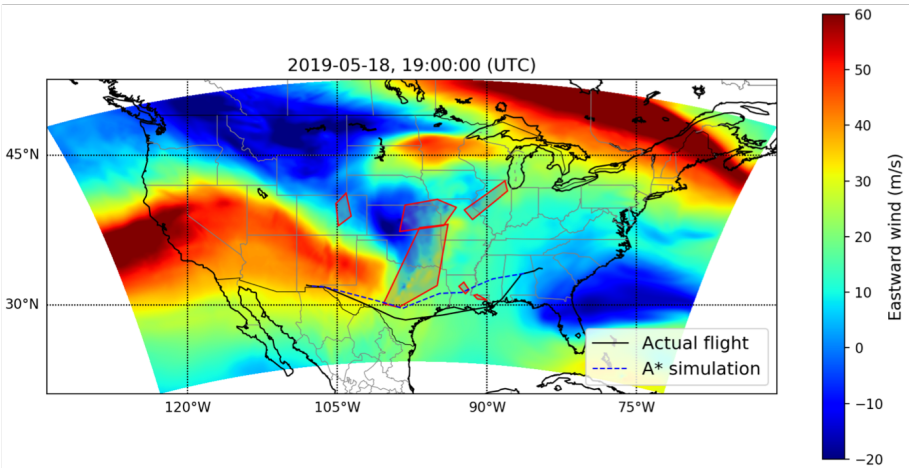
Figure 2.5: Example routes generated by the PARTNER (Reprinted from [33])

While the PARTNER would provide a parametric real-time en-route flight path optimization that was tested preliminarily as a proof of concept, it is important to emphasize that the main issue with the PARTNER is that the tool does not incorporate weather forecast information into the optimization framework and instead use demo weather data (e.g., simple rectangular box as a no-fly zone). This indicates that the PARTNER has some limitations in its applicability for realistic flight use cases.

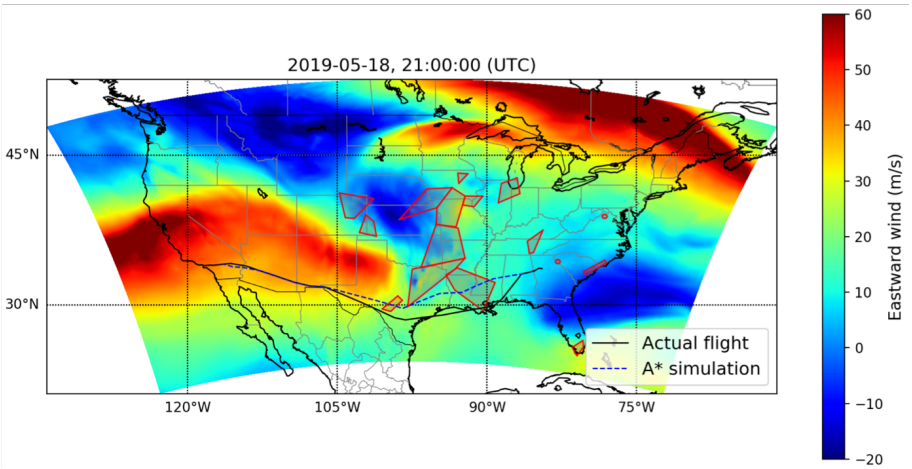
In response to this concern, in 2019, the ASDL research group improved the fidelity of the PARTNER by automatically fusing various weather forecast products such as the National Oceanic and Atmospheric Administration (NOAA)'s winds and the Aviation Weather Center (AWC)'s convective Significant Meteorological Information (SIGMET) polygons and by implementing the A* search algorithm with a network that considers the U.S. airspace infrastructure [34]. The upgraded version of the PARTNER was named the Real-time Trajectory OPtimization (RTOP). Figure 2.6 shows an example route generated by the RTOP-v1. In this case, the RTOP-v1 generated the simulated optimal route shorter than the actual historical flight (i.e., DL1854).



(a) Real flight vs. Simulation at 2019-05-18 17:00 UTC



(b) Real flight vs. Simulation at 2019-05-18 19:00 UTC



(c) Real flight vs. Simulation at 2019-05-18 21:00 UTC

Figure 2.6: Flight trajectory comparison between DL1854 and RTOP-v1

While the RTOP-v1 established a framework that enables real-time flight path optimization based on weather forecast products that are currently used in the aviation industry, there were some potential shortcomings identified when the RTOP-v1 was validated with real flight scenarios: First, the RTOP-v1 provided the flight trajectory in which aircraft flew unnecessary detours due to limitations in the accuracy of the AWC convective SIGMET data [35]. Second, the RTOP-v1 was so conservatively constrained by the U.S. airspace infrastructure that it generated an optimal flight path only with user-supplied information. This research attempts to resolve those potential shortcomings by leveraging various machine learning and optimization techniques (i.e., RTOP-v3). Additional details will be discussed in Chapter 4 and Chapter 5.

CHAPTER 3
BACKGROUND

Prior to identifying research questions and hypotheses to be addressed in Chapter 4, this chapter provides relevant background information on machine learning and optimization techniques used for the proposed methodology presented in Chapter 5. The intent of this chapter is not to provide a comprehensive discussion of each technique but rather to give enough background for readers who may have little knowledge of the techniques.

3.1 Artificial Neural Network

The Artificial Neural Network (ANN) has been widely used in numerous engineering problems (e.g., computer vision, surrogate modeling, and natural language processing) over the past years and now routinely appears in a variety of industries. In general, the ANN is defined as a non-linear function that is adjusted by weight parameters during a training process with the aim of creating a prediction model. The process works by mapping a set of input variables in an input layer to a set of responses in an output layer through a set of hidden layers. The ANN typically entails the following model structure as shown in Figure 3.1: 1) an input layer to receive the model input data, 2) a hidden layer that is considered as a computational engine for finding the best weight parameters by an iterative procedure, and 3) an output layer to make a prediction.

In fact, understanding the structure of the human brain as shown in Figure 3.2 is necessary to have a solid background of how the ANN works because the ANN is based on a theory of neurons in the human brain. The process of how our brain works is illustrated as follows: 1) signals are received from dendrites, 2) a nucleus makes a summation for all incoming signals, 3) an axon reacts if the sum of signals is enough, and 4) the outgoing signal is used as another input for other neurons. The basic idea of the ANN is to mimic

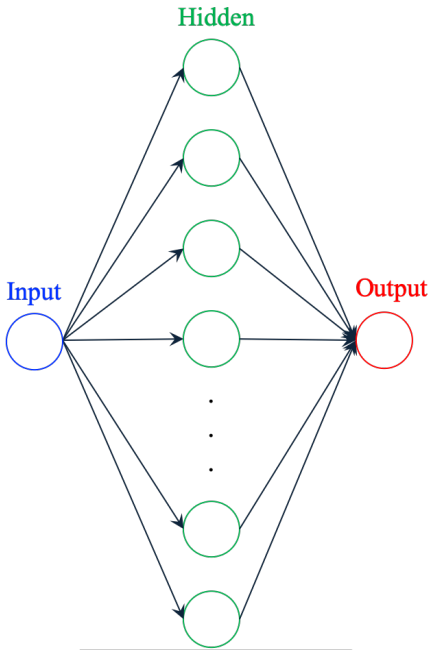


Figure 3.1: Notional diagram of the ANN model structure

this process in a way that it receives a list of weighted input signals and gives an output signal if the sum of the input signals reaches a certain value. This is the first mankind's mathematical model of the ANN introduced by S. McCulloch and W. Pitts in 1943 [36].

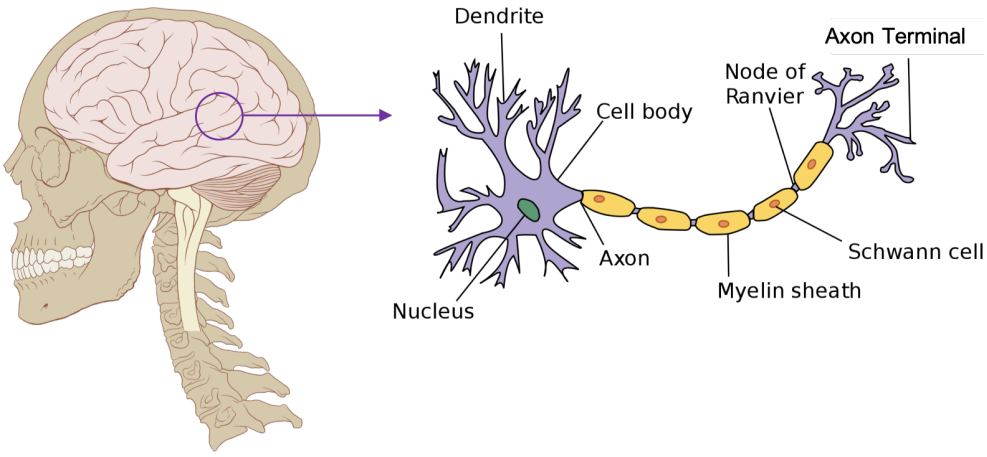


Figure 3.2: Structure of the human brain (Adapted from [37])

In 1958, Frank Rosenblatt proposed the new concept, called the Perceptron, which was

a more generalized computational model [38] than the model invented by S. McCulloch and W. Pitts in a way that weights of the mathematical model are learned over time. The concept of the Perceptron introduced functionality that adjusts weight parameters during a training process by introducing activation functions such as a sigmoid function having a characteristic S-shaped curve as shown in Figure 3.3.

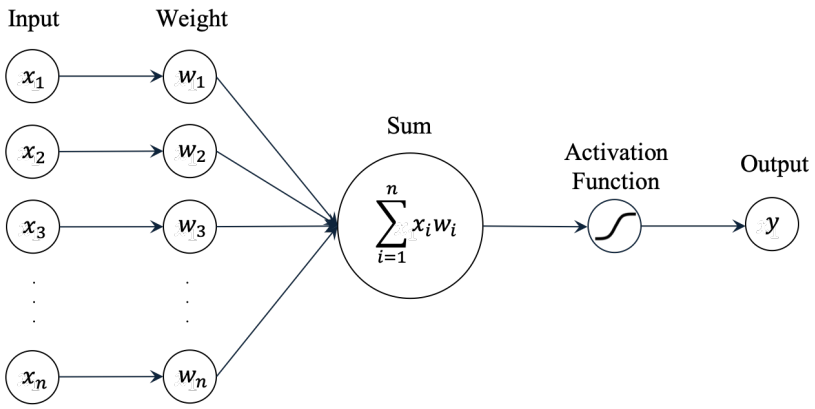


Figure 3.3: Schematic of the Perceptron

Although Frank Rosenblatt's idea stunned the world at the time, the Perceptron's popularity rapidly decreased as Marvin Minsky and Seymour Papert published their paper in 1969 [39] where they mathematically validated that the Perceptron could not resolve XOR problems [40] as it was defined as a simple linear function. This invoked the first dark age of the ANN, namely the first ANN winter. The first ANN winter was terminated by the idea called the Multi Layer Perceptron (MLP) [41] proposed by David E. Rumelhart and Geoffrey E. Hinton in 1986. The idea enabled them to address the XOR problems by drawing multiple decision boundaries with the structure shown in Figure 3.4. The MLP structure entails several different layers (i.e., input layer, hidden layer, and output layer) with multiple artificial neurons and each neuron represents the Perceptron. The MLP has two fundamental steps called Forward Propagation and Back Propagation as illustrated follows:

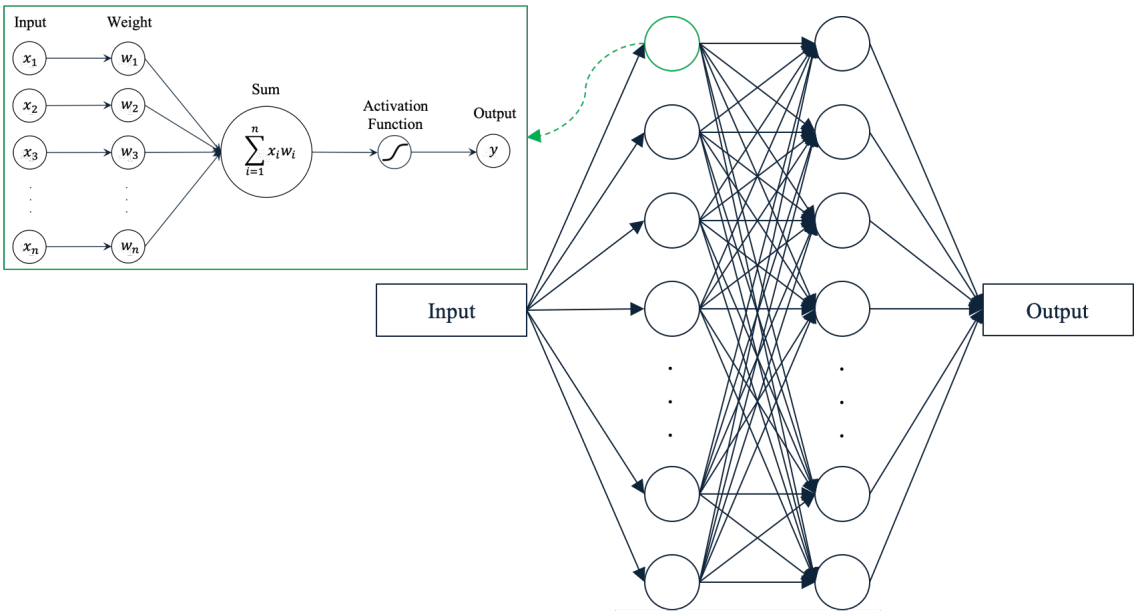


Figure 3.4: Structure of the MLP

Forward Propagation

Given that the MLP structure shown in Figure 3.5 is provided as an example, the process starts by randomly initializing the weight parameters (i.e., $\theta_{ij}^{(l)}$) of the hidden layers. It is important to note that initialization values should not be defined by zero vector; otherwise, hidden units become identical in the hidden layer. Once the weight parameters are initialized, the next step is to implement forward propagation in order to estimate an output response from a hypothesis function (i.e., $h_\theta(x)$) with a set of input values (i.e., x_i). For example, an output response at the first iteration is mathematically derived with an input and initial weight parameters as follows:

$$h_\theta(x) = h_1^{(4)} = g(\theta_{10}^{(3)} \cdot b_0^{(3)} + \theta_{11}^{(3)} \cdot h_1^{(3)} + \theta_{12}^{(3)} \cdot h_2^{(3)}) \tag{3.1}$$

$$h_1^{(3)} = g(\theta_{10}^{(2)} \cdot b_0^{(2)} + \theta_{11}^{(2)} \cdot h_1^{(2)} + \theta_{12}^{(2)} \cdot h_2^{(2)}) \tag{3.2}$$

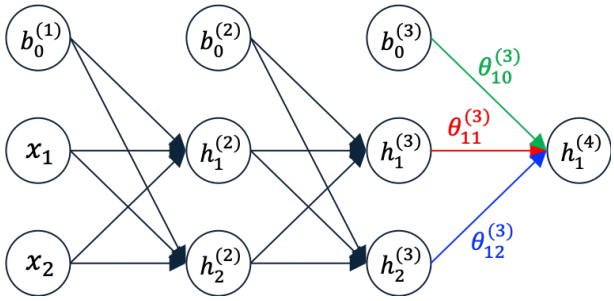


Figure 3.5: Simple MLP structure as an example

$$h_2^{(3)} = g(\theta_{20}^{(2)} \cdot b_0^{(2)} + \theta_{21}^{(2)} \cdot h_1^{(2)} + \theta_{22}^{(2)} \cdot h_2^{(2)}) \tag{3.3}$$

$$h_1^{(2)} = g(\theta_{10}^{(1)} \cdot b_0^{(1)} + \theta_{11}^{(1)} \cdot x_1 + \theta_{12}^{(1)} \cdot x_2) \tag{3.4}$$

$$h_2^{(2)} = g(\theta_{20}^{(1)} \cdot b_0^{(1)} + \theta_{21}^{(1)} \cdot x_1 + \theta_{22}^{(1)} \cdot x_2), \tag{3.5}$$

where $h_i^{(j)}$ represents activation of unit i in layer j, $\theta^{(j)}$ refers to weight factor function that is mapping from layer j to layer $j+1$, and $b_0^{(j)}$ means a bias node in layer j. A single bias node is needed per layer because it helps a learning process by allowing an activation function to shift to either left or right as necessary. An activation function g can be Sigmoid, Gaussian, or TanH function. In fact, activation functions are significantly important for a neural network structure as the functions introduce non-linear properties to the network system. The scaled version of the Sigmoid function defined by Equation 3.6, namely TanH, is typically employed.

$$g(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{3.6}$$

The output response at the first iteration is estimated by making a summation of all input values and initialized weight values. The estimated response is then compared with

the real output value to measure the accuracy of the hypothesis function (i.e., $h_\theta(x)$). A squared error function (i.e., $J(\theta)$) defined by Equation 3.7 is typically utilized to measure the accuracy. It is important to note that the process is iterated with different training datasets (i.e., $x^{(i)}$) after the first iteration.

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y}_i - y_i\right)^2, \tag{3.7}$$

where $\hat{y}_i$ represents a predicted value from forward propagation and y_i refers to the actual value given from the real output; thus, the error function basically means the difference between the predicted value and the actual value. If the error function provides a value larger than user-defined convergence criteria, the weight parameters are updated using the Back Propagation algorithm.

Back Propagation

David E. Rumelhart and Geoffrey E. Hinton proposed the algorithm called Back Propagation to update weight parameters in the MLP structure. The key idea of the algorithm is to update the weight parameter where the error is minimized. The mathematical equation is typically formulated by Equation 3.8.

$$W_{new} = W_{old} + \alpha \frac{\partial J(\theta)}{\partial W_{kj}}, \tag{3.8}$$

where α represents the learning rate. The chain rule is used to calculate the derivative as defined in Equation 3.9. Once the Back Propagation algorithm completes updating all the weight parameters, the Forward Propagation algorithm starts with the new weight parameters. This process is repeated until it is converged.

$$\alpha \frac{\partial J(\theta)}{\partial W_{kj}} = \frac{\partial J(\theta)}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial x_j} \frac{\partial x_j}{\partial W_{kj}} \tag{3.9}$$

In addition to the success of the concept of the MLP with forward and back propagation,

George Cybenko published the Universal Approximation Theorem in 1989, claiming that a two-layer network can uniformly approximate any continuous function with arbitrary accuracy if 1) the network has enough hidden units and 2) the output layer is defined with a linear activation function [42]. Consequently, the ANN-based methods regained popularity from the other techniques at the time. In summary, Figure 3.6 [43] shows milestones in the development of the ANN. It is important to note that deep learning (i.e., more than two hidden layers) techniques are recently introduced to solve complex problems but this research does not consider the deep learning techniques because the techniques typically require high computational costs so that they may not be feasible in real-time analysis.

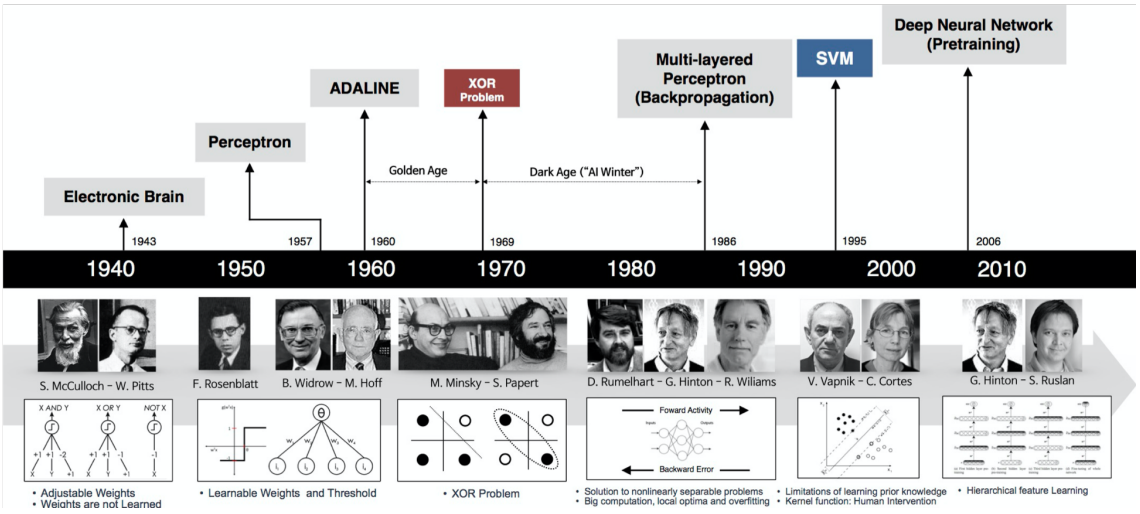


Figure 3.6: Milestones in the development of the ANN (Reprinted from [43])

This research particularly employed the MLP as a regression function with the aim of finding the best weight parameters minimizing an error between predicted and target wind values. The effective MLP model was determined by the choice of hyperparameters. Additional details will be discussed in section 5.2.

3.2 Support Vector Machine

The Support Vector Machine (SVM), proposed by Corinna Cortes and Vapnik in 1993 and published in 1995 [44], is one of the most robust supervised machine learning algorithms used for classification and regression problems. The SVM works by finding a hyperplane that maximizes the margin such that 1) data points are distinctly separated for a classification problem and 2) data points deviate less than the required accuracy from real values for a regression problem. The core of the SVM algorithm is to construct a hyperplane and the hyperplane is established by formulating an optimization problem where the margin is maximized. Figure 3.7 notionally delineates a concept of the Support Vector Regression (SVR) used for a regression problem. Here, the vectors that construct the hyperplane are called support vectors.

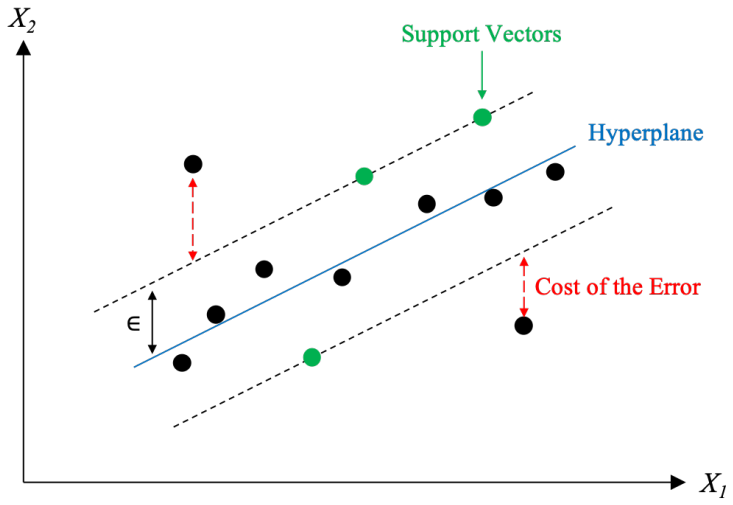


Figure 3.7: Notional sketch of the SVR

The beauty of the SVM is that it can transform an original feature space (e.g., given that the original feature space is not linearly separable) to a high dimensional space by deploying a non-linear kernel trick such as the Radial Basis Function (RBF) as notionally illustrated in Figure 3.8. It is worth mentioning that kernel functions have to be mathemat-

ically validated (i.e., symmetric and positive semi-definite) for use in the SVM algorithm.

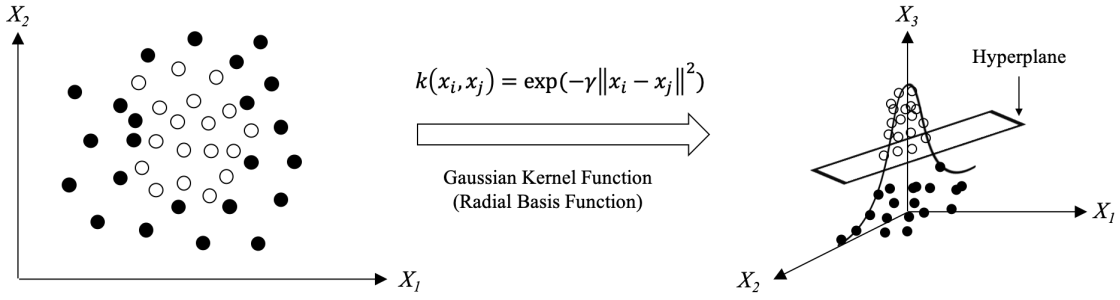


Figure 3.8: Notional sketch of deploying a non-linear kernel function

Given that a kernel function is valid, the choice of the kernel function in the SVM algorithm naturally leads to considering hyperparameters related to the kernel function. For example, the RBF kernel function typically requires to specify two hyperparameters (i.e., penalty parameter and RBF-related parameter) as the SVM algorithm is sensitive to the hyperparameters. The penalty parameter, which is sometimes referred to as a regularization parameter, balances between correct and incorrect data points with the penalty values in the objective function of the optimization problem. The RBF-related parameter, called a gamma parameter, basically controls the influence of data points. More specifically, the SVM algorithm is too constrained as the gamma parameter becomes small; thus, the algorithm cannot capture the complexity of data points. On the other hand, if the gamma parameter becomes large, the SVM algorithm is affected by an over-fitting issue.

This research employed the SVM as a regression function to obtain continuous wind information in an entire two-dimensional U.S. airspace. The choice of the SVM algorithm in this research was primarily determined based on the fact that the MLP has the potential to be trapped in a local minimum as the weight optimization process in the hidden layers is considered as a convex problem. Furthermore, this research evaluated a few valid kernel functions and utilized the RBF kernel function to transform the original feature space. Additional details will be discussed in section 5.2.

3.3 Gaussian Process

The Gaussian Process (GP), which was originally introduced by Georges Matheron [45] as a geostatistical estimation method, is a supervised machine learning algorithm widely utilized in a general regression problem. The obvious upside of the GP algorithm is that its prediction is probabilistic in a way that it provides meaningful uncertainty bounds along with the predictions. However, it must be noted that the algorithm may require high computational costs when it makes a prediction with numerous datasets.

The GP algorithm internally employs the kernel trick during a training process to define the covariance of a prior distribution over the output function (e.g., wind prediction). Kernel functions are an imperative element for the GP algorithm especially for determining the shape of a prior distribution [46]. The most widely utilized kernel function for the GP algorithm is the squared exponential function [47] and its form is hyperparameters $\theta = [\theta_0,\, \theta_1,\, \beta^{-1}]$ are learned by maximizing the log marginal likelihood of the output responses given input datasets as defined in Equation 3.10.

$$\theta_{max} = \arg\max_{\theta}\{\, \ln p(y_{1:N} \mid x_{1:N}, \theta)\, \} \tag{3.10}$$

The choice of the kernel function naturally leads to considering the determination of hyperparameters. In particular, the hyperparameters of the GP algorithm are optimized during a training process by maximizing the log marginal likelihood based on an optimizer (e.g., Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm [48]). This research used the GP algorithm with the squared-exponential kernel function to create a non-linear probabilistic regression model of wind data. Additional details will be discussed in section 5.2.

3.4 Density-Based Spatial Clustering of Applications with Noise

The Density Based Spatial Clustering of Applications with Noise (DBSCAN), one of the most popular unsupervised machine learning techniques, is a data clustering algorithm in-

troduced by Martin Ester in 1996 [49]. The DBSCAN clusters data points by grouping points that are located close to each other in a region and separating the points from a set of data points whose nearest neighbors are located far away from the region. The DBSCAN algorithm is different from other traditional clustering algorithms such as the K-means algorithm [50] and the EM algorithm [51] in a way that 1) it introduces the notion of noise points and 2) it does not require the number of clusters before running the algorithm. This indicates that the DBSCAN algorithm requires minimum domain knowledge and generates an arbitrary shape of clusters.

Given a set of data points in a dimensional space, the DBSCAN algorithm starts by choosing an arbitrary point that has not been visited. At the starting point, the algorithm retrieves neighborhood information based on the hyperparameter of the DBSCAN algorithm called $Epsilon$. More specifically, a point is considered as neighbors if the distance between the point and the starting point is less than $Epsilon$. On the other hand, a point is not considered as neighbors if the distance between the point and the starting point is greater than $Epsilon$. It is important to note that a large portion of data points is considered outliers if $Epsilon$ is chosen too small; while the majority of data points are in the same cluster if $Epsilon$ is chosen too large.

Once neighborhood information of the starting point is identified, the next step is to identify types of data points based on another hyperparameter of the DBSCAN algorithm called $minPts$ corresponding to the number of points. More specifically, the DBSCAN algorithm specifies three different types of data points during a clustering process as shown in Figure 3.9: 1) a point is defined as a core point if the number of points within the radius of $Epsilon$ is larger than $minPts$, 2) a point is defined as a border point if the number of points within the radius of $Epsilon$ is fewer than $minPts$ but the point is in the neighborhood of a core point, and 3) a point is defined as a noise point if it is not either core or border point; thus, the point is not assigned to a cluster.

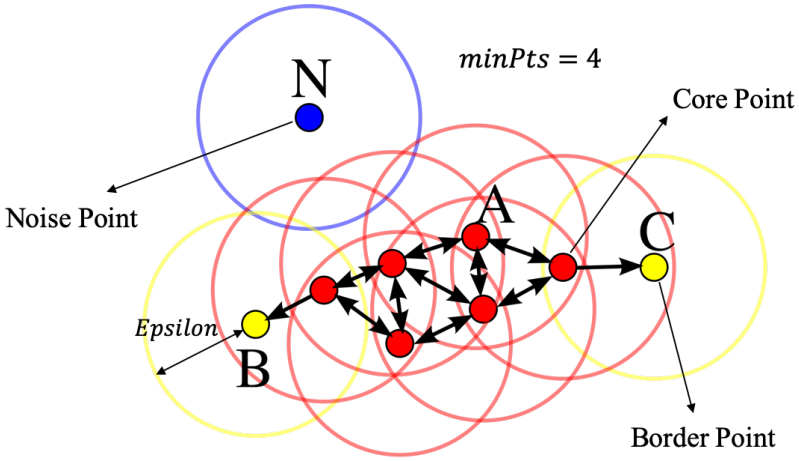


Figure 3.9: Types of data points in the DBSCAN (Adapted from [52])

After identifying types of data points, the DBSCAN algorithm adds all data points within the radius of $Epsilon$ of the starting point and starts formulating a cluster group with the data points. The algorithm repeats the process (i.e., retrieving neighborhood information and identifying types of data points especially for the neighbors) for all of the new points that have been added to the cluster group. The recursive process is repeated until density connected points are completely found. The density connected point (i.e., border point) is defined as a point that is directly reachable via other specified core points. Once the algorithm completes formulating the current cluster group, a new data point that has not been visited is selected and processed with the same procedure mentioned above. This process is repeated until all data points are marked as visited.

This research used the DBSCAN algorithm to cluster a set of scattered points in U.S. airspace consisting of different radar reflectivity values. The DBSCAN algorithm was particularly employed for this research because 1) weather typically requires minimum domain knowledge (i.e., it is not necessary to require the number of clusters before running the algorithm) and 2) weather normally contains noisy and irregular datasets (i.e., it is robust to outliers). Additional details will be discussed in section 5.3.

3.5 K-Nearest Neighbor

The K-Nearest Neighbor (K-NN), which was first developed by Evelyn Fix and Joseph Hodges in 1951 [53], is an instance-based learning algorithm (i.e., lazy learning algorithm) that does not generate a model but construct hypotheses directly from training data [54]. The K-NN algorithm can be used for classification and regression problems in a way that 1) an output is classified by counting votes from its neighbors in a classification problem and 2) an output is estimated by considering the average of neighbor values in a regression problem. Figure 3.10 notionally shows how the K-NN algorithm works in the classification problem. In this example, the test point (i.e., green dot) is most likely to be classified as a red triangle if K is equal to three (i.e., solid black circle line) because there are two red triangles and one blue square in the circle.

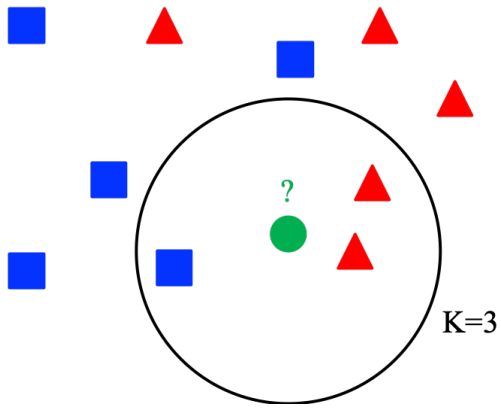


Figure 3.10: Example of the K-NN classification

Details of the process is as follows: The first step is to specify the hyperparameter of the K-NN algorithm called K. For example, if K is equal to 10, the algorithm starts searching 10 data points close to the test point. To identify the points close to the test point, the K-NN algorithm typically utilizes the Euclidean distance as defined in Equation 3.11 to calculate the distance between two points. It is important to note that the K-NN algorithm

may require high computational costs if the algorithm must address numerous data points.

$$Distance(point1, point2) = \sqrt{\sum_{i=1}^{n} (point1_i - point2_i)^2} \tag{3.11}$$

Given that the problem is defined as a classification problem, the second step is to determine a class membership based on class information of the selected points. Normalization of data points is generally recommended to reduce an error associated with calculating the distance between data points. Min-Max normalization defined as Equation 3.12 is typically employed.

$$X_{normalization} = \frac{X_{original} - min(x)}{max(x) - min(x)} \tag{3.12}$$

While the K-NN algorithm can be implemented very easily, it is important to note that the algorithm is computationally expensive especially if it must account for calculating distance with many data points. In response to this concern, there have been many attempts to resolve computational issues. Ball tree [55] and K-Dimensional tree [56] are the most representative techniques to address the issue. The techniques basically answer the following question: Given N points in a dimensional space, how can a pair of points with the smallest Euclidean distance be found more efficiently? The reason why the K-Dimensional tree is better than the naive approach (e.g., Brute-Force) is that the K-Dimensional tree neglects to calculate some points far from the test point. Figure 3.11 delineates a brief explanation of how the K-Dimensional tree works for finding a pair of points with the smallest distance.

As can be seen, it begins by choosing a root node and splits the data points into two groups. In this case, it selects the median value in the x-axis as the root node. It recursively constructs the K-Dimensional tree in both left and right half-spaces by picking other points to split. In this case, it selects the median value in the y-axis from each half-space. It continues the construction to completion by repeating the process that selects the point to split from either the x-axis or y-axis. At this stage, the K-Dimensional tree is completely

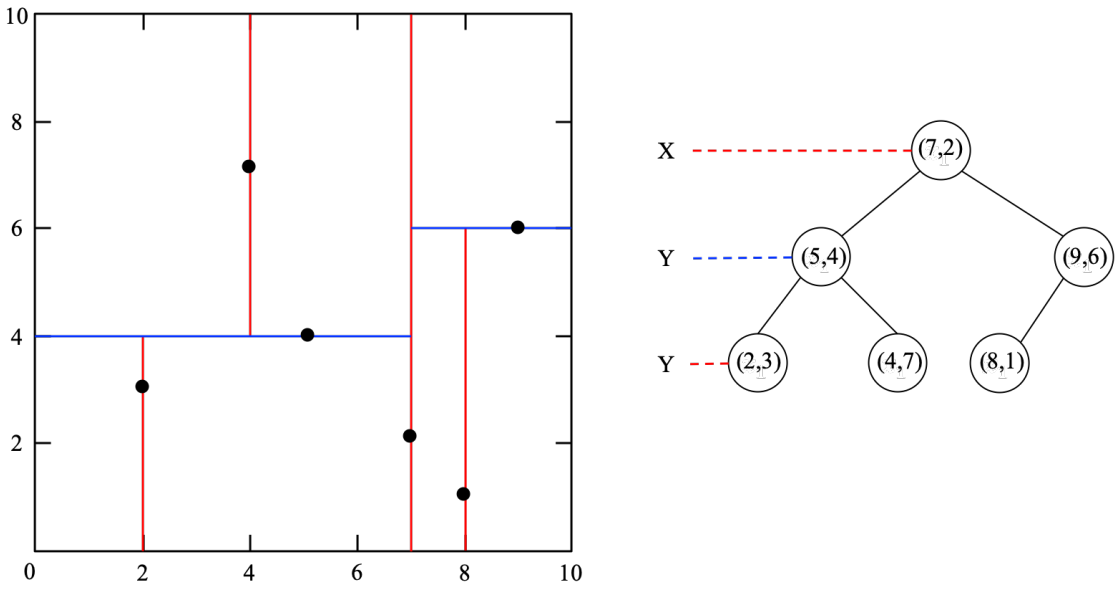


Figure 3.11: Example of the K-Dimensional tree decomposition

constructed. The next step is to identify which point in the K-Dimensional tree is closest to the test point. Once the point is identified, it becomes the current guess. It is important to note that the current guess does not represent the final guess. The final guess is achieved by drawing a circle with a radius that is connected by the line of two points. It is necessary to update the current guess if there is a point in the circle. If there is not a point in the circle, the current guess becomes the final guess.

While the K-Dimensional tree works well with a low dimensional space, it is important to note that the K-Dimensional tree is not effective for a high dimensional space (i.e., the curse of dimensionality). The Ball tree, another space partitioning data structure for organizing points, is introduced to resolve the curse of dimensionality. The key difference between two techniques is that the K-Dimensional tree partitions data points with Cartesian axes; while the Ball tree, which is sometimes referred as the advanced K-Dimensional tree, partitions data points with a nested set of hyperspheres. In this way, the Ball tree works better than the K-Dimensional tree for high dimensionality.

This research particularly employed the Ball tree because a designated points-based

flight path optimization model presented in section 5.4 needed an efficient way that finds the nearest neighbor waypoints, especially when it is required to generate additional free-flight routes. Additional details will be discussed in section 5.4.

3.6 A* Search Algorithm

The A* search algorithm, which was first published by the Stanford Research Institute in 1968, is a path search algorithm in which the objective is to identify the shortest path from a specific starting node in the network to a specific end node. The A* search algorithm is sometimes referred to as an informed search algorithm because it is typically formulated with pre-determined information (e.g., weighted network graph), meaning that the algorithm provides the shortest path only within the pre-determined network graph.

The A* search algorithm works by combining two different types of cost functions (i.e., $G(n)$ and $H(n)$), associated with calculating the distance, to balance between accuracy and speed in the algorithm. $G(n)$, called an exact cost function, represents exact costs from a starting point to n, whereas $H(n)$, called a heuristic cost function, represents estimated costs from n to an end point. The objective of the A* search algorithm is to minimize the sum of these two cost functions (i.e., $F(n)$) as shown in Equation 3.13.

$$F(n) = G(n) + H(n) \tag{3.13}$$

The secret to the A* search algorithm's success is based on the fact that the algorithm simultaneously considers the points close to a starting point as well as the other points close to an end point; thus, it performs vertex expansion efficiently. For this reason, the A* search algorithm is typically considered as one of the most popular algorithms for a pathfinding problem. Another important aspect of the A* search algorithm is that the algorithm is guaranteed to find an optimum with a reasonable computation cost once the properties (i.e., completeness, admissibility, consistency, and optimal efficiency) of the algorithm are

satisfied [57]. Details of the properties are summarized as follows:

Completeness

Given that a graph does not have negative edge weights, the A* search algorithm will always find a solution if there is the solution.

Admissibility

The solution generated by the A* search algorithm is optimum if the heuristic cost function is admissible. Equation 3.14 shows a mathematical definition of admissibility in the A* search algorithm.

$$Heuristic(x, goal) \leq Actual(x, goal) \tag{3.14}$$

Consistency

A heuristic cost function is consistent if it satisfies an inequality as defined in Equation 3.15. Figure 3.12 shows a notion of consistency in the A* search algorithm.

$$|Heuristic(x, goal) - Heuristic(y, goal)| \leq Actual(x, y) \tag{3.15}$$

Optimal efficiency

Given a pathfinding problem, the A* search algorithm is efficient compared to other similar graph-based search algorithms once the A* search algorithm is not only admissible but also consistent. In other words, no algorithms can find the shortest path between a pair of nodes on the network in a smaller computational cost compared with the A* search algorithm.

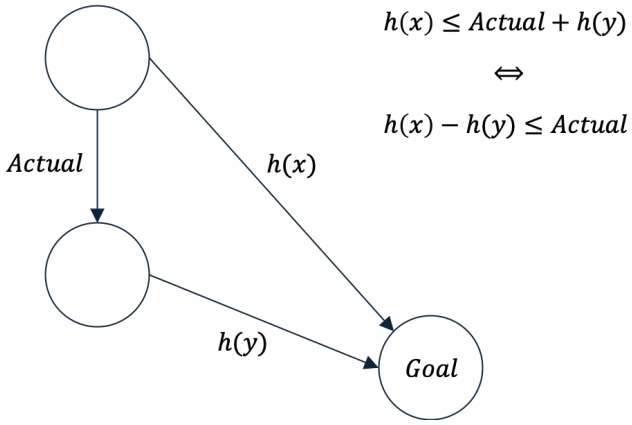


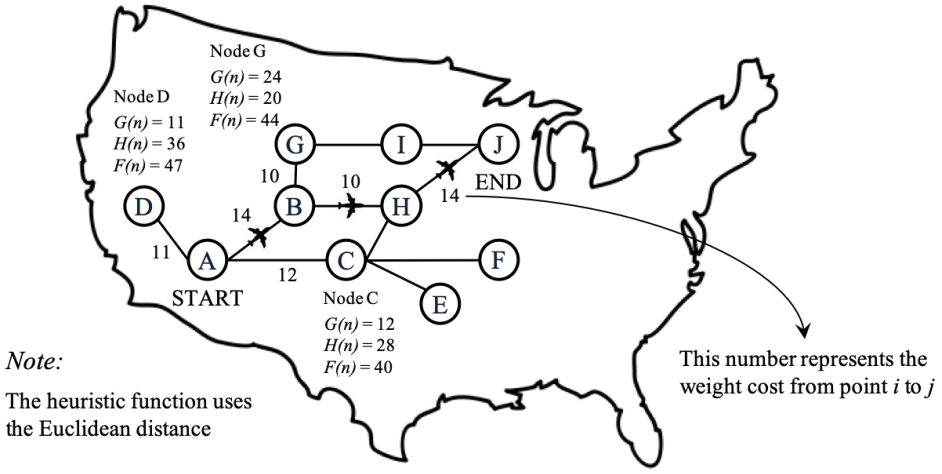
Figure 3.12: Notion of consistency in the A* search algorithm

A canonical example of how the A* search algorithm performs is provided in Figure 3.13. More specifically, the A* search algorithm starts by specifying two different sets that are called OPEN and CLOSE respectively. The OPEN set is designed to contain nodes that are supposed to be examined. The CLOSE set is designed to contain nodes that have already been examined. Initially, the OPEN set contains only one element, namely the starting point, and the CLOSE set is empty. The initial cost is then computed by Equation 3.13. It is important to note that the canonical example utilized the Euclidean distance as the heuristic cost function.

After calculating the initial cost with the starting point, the starting point is removed from the OPEN set and added to the CLOSE set. At the same time, the OPEN set keeps track of nodes that can be visited from the starting point and adds them to the set. The cost for each node is computed by Equation 3.13. Once the algorithm completes computing the cost for each node, it identifies which node has the lowest cost. The node with the lowest cost becomes a new starting point and the recursive process is repeated. In general, a priority queue [58] is recommended to use in the A* search algorithm because there seems a mechanism that repeatedly pulls out the best node (i.e., the lowest cost in this case) in the OPEN set.

The A* search algorithm also requires two additional sets that are called CURRENT

and PARENT respectively. The sets are necessary for the algorithm to keep track of the previous node from the current node in case the algorithm needs to accumulate the total cost along with the optimal nodes. It is important to note that the A* search algorithm is designed to choose a node with the smallest heuristic cost function value when two candidate nodes have exactly the same cost values. The A* search algorithm completes its process if the node in the CURRENT set is the goal.



Iteration 1. CURRENT = [A], OPEN = [A], CLOSE = [ ], PARENT = [ ], $G(n) = 0$, $H(n) = 36$, $F(n) = 36$

Iteration 2. CURRENT = [B], OPEN = [B,C,D], CLOSE = [A], PARENT = [A], $G(n) = 14$, $H(n) = 22$, $F(n) = 36$

Iteration 3. CURRENT = [H], OPEN = [H,C,G,D], CLOSE = [A,B], PARENT = [B], $G(n) = 24$, $H(n) = 14$, $F(n) = 38$

Iteration 4. CURRENT = [J], OPEN = [J,C,D,G], CLOSE = [A,B,H], PARENT = [H], $G(n) = 38$, $H(n) = 0$, $F(n) = 38$

Figure 3.13: Canonical example of how the A* search algorithm works

A high-level summary of the process is presented in Algorithm 1. There may be a number of different ways to implement the A* search algorithm; however, it is worth mentioning that Algorithm 1 is directly employed in this research to find an optimal flight route. Additional details will be discussed in section 5.4.

Algorithm 1 Pseudocode of the A* search algorithm

```
Initialize lists (OPEN, CLOSE, CURRENT, PARENT)
Specify Start and Goal nodes
Add the Start node to the CURRENT list
Add neighbor nodes connected to the Start node (n) to the OPEN list
Evaluate cost for the current state
Add the current state to the CLOSE list
function A STAR(Start, Goal)
    while OPEN is not empty and CURRENT is not the Goal node do
        for n = 1, 2, ..., N do
            if n[i] is already in the CLOSE list then
                Skip n[i]
            else if n[i] is not in the CLOSE list then
                if n[i] is already in the OPEN list then
                    if A new cost is lower than the previous cost then
                        Remove the previous case
                        Evaluate cost for the n[i]
                        Add n[i] to the OPEN list
                    else if A new cost is larger than the previous cost then
                        Skip n[i]
                    end if
                else if n[i] is not in the OPEN list then
                    Evaluate cost for n[i]
                    Update the OPEN list with n[i]
                end if
            end if
        end for
        Update the CLOSE list with the minimum cost case
        End of state evaluation cycle
    end while
    End of search cycle
end function
Accumulate the total cost using both CLOSE and PARENT lists
```

CHAPTER 4
RESEARCH FORMULATION

This research started by observing two potential issues related to current in-flight re-planning systems identified from the interview with the pilots. The identified issues presented in Chapter 1 are as follows: First, current in-flight re-planning systems are not fully automated; thus, pilots today perform some portions of the in-flight activities manually. Second, weather forecasts used for current in-flight re-planning systems are not always accessible in a timely manner. This motivated to establish the objective of this research as follows:

> *Research Objective: Develop an automated framework that performs in-flight re-planning continuously with the latest weather information sets available*

The research objective naturally led to review previous work related to the aim of improving a continuous flight re-routing process. The literature review presented in Chapter 2 outlined several attempts to develop capabilities improving in-flight re-planning systems under the following areas: 1) flight management system, 2) ground-based flight planning framework, 3) electronic flight bag, and 4) machine learning-based flight route prediction. The literature search identified research gaps as follows:

- Research Gap 1: A flight management system reduces the workload of the flight crew by automating a variety of in-flight activities but some parts in the flight planning functionality are still manual input.
- Research Gap 2: A ground-based flight planning framework has mostly automated the task of flight planning for flight dispatchers but the framework does not address the connectivity issue of co-constructing knowledge between pilots and flight dispatchers, leading to disapproved requests resulting in inefficient communication.

- Research Gap 3: The most widely used aviation-related applications implemented in the electronic flight bag have proved their capabilities in certain application areas; however, the applications may not be feasible for a real-time flight path optimization framework that uses the latest weather information to continuously update flight routes.
- Research Gap 4: Deep learning techniques are prevalently adopted in the aviation industry; however, these techniques may not be feasible for a real-time flight path optimization framework due to high computational costs.

The aforementioned research gaps helped this research define the Overarching Research Question as follows:

Overarching Research Question: How can a cockpit-based real-time flight path optimization framework, which automatically performs in-flight re-planning continuously with the latest weather information, be developed?

To answer the Overarching Research Question, this chapter will pose research problems (i.e., supervised machine learning-based wind regression, unsupervised machine learning-based convective weather prediction, and designated points-based flight path optimization) that are specifically formulated to answer the research questions as follows:

Research Question 1: How can the capability of providing weather information accurately and continuously to a cockpit-based real-time flight path optimization framework within a specified time be developed?

Research Question 2: How can the capability of providing simulated optimum flight routes automatically to a cockpit-based real-time flight path optimization framework be developed?

The remainder of this chapter is organized as follows. The first and second sections formulate research problems to answer the Research Question 1. The third section presents a research problem to answer the Research Question 2. The last section summarizes all of the research statements introduced in Chapter 4.

4.1 Supervised Machine Learning-based Wind Regression

Many research groups such as the NOAA have been committed to developing numerical models for weather forecasts. The most well-known numerical weather models are tabulated in Table 4.1. Two representative weather forecast models are the Global Forecast System (GFS) developed by the NOAA and the European Center for Medium-Range Weather Forecast (ECMWF) developed by the European Centre (CoE), respectively called the American model and the European model.

Table 4.1: The most well-known numerical weather models

Model	Organization	Spatial Resolution	Temporal Resolution	Open Source	Region
MERRA-2	NASA	27 km	3 hours	Yes	Global
NAM	NOAA	12 km	3 hours	Yes	USA
RAP	NOAA	13 km	1 hour	Yes	USA
GFS	NOAA	13 km	1 hour	Yes	Global
HRRR	NOAA	3 km	1 hour	Yes	USA
ECMWF	CoE	9 km	1 hour	No	Global

With the development of numerical weather models, many research groups have been able to predict weather patterns and trends. In particular, as the weather models are based on physical equations (e.g., Navier-Stokes equations) [59], the models predict wind values that are relatively accurate than other parameters in the models. For this reason, pilots typically receive wind information predicted by the models and use the forecast to not

only calculate how much fuel is required for a flight but also optimize flight routes by seeking favorable winds. One potential issue, however, is that the models provide relatively sparse (i.e., discrete) wind information in both space and time, potentially leading to an inaccurate calculation of fuel consumption. Figure 4.1 notionally illustrates the reason why it is important to create a reliable model that provides continuous wind information with respect to a four-dimensional flight trajectory (i.e., timestamp, altitude, latitude, and longitude).

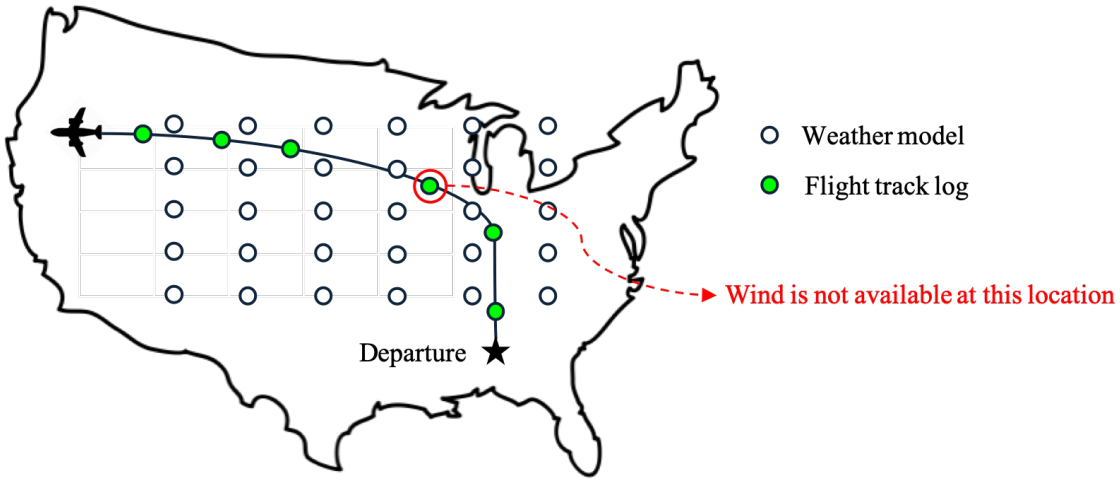


Figure 4.1: Notional sketch of the necessity for continuous wind information

The most common approach for obtaining continuous wind information in current in-flight re-planning systems is via a linear interpolation method. For example, the FMS uses wind information forecasted by numerical weather models and interpolates wind values at any altitude or waypoint as needed [15]. The TAP developed by NASA employs a linear interpolation method to provide continuous Rapid Refresh (RAP) wind data to the in-flight re-planning system [21]. The RTOP-v1 developed by the ASDL utilizes a linear interpolation method with the High Resolution Rapid Refresh (HRRR) wind data to calculate total travel time [34]. The Aviation Environmental Design Tool (AEDT) developed by the Federal Aviation Administration (FAA) uses a linear interpolation method with the Modern-Era

Retrospective analysis for Research and Applications-2 (MERRA-2) wind data to calculate fuel consumption in the simulation environment [60].

A linear interpolation method may be appropriate for obtaining continuous wind information at cruising altitudes given that winds do not change dramatically over time at these altitudes. However, it is important to note that the linear interpolation method may have some limitations in predicting wind patterns where non-linear behavior is dominant. Based on the aforementioned trends, the following observation can be claimed:

Observation 1.1: The most common approach for obtaining continuous wind information in current in-flight re-planning systems is via a linear interpolation method; however, the linear interpolation method may have some limitations in predicting wind patterns where non-linear behavior is dominant.

Based on the Observation 1.1, the following research question can be constructed:

Research Question 1.1: How can continuous wind information be provided more accurately (i.e., lower prediction error) other than using the current linear interpolation method?

Recently, with the advent of AI, many machine learning algorithms have been widely used with a data-driven approach to enhance the level of understanding of wind phenomena. For example, Ashish Kapoor et al. [61] developed a probabilistic graphical model with a data-driven approach to estimate the wind and aircraft true airspeed. M.A. Mohandes et al. [62] compared two different machine learning algorithms for wind speed prediction. Ronay Ak et al. [63] combined different machine learning approaches for short-term wind speed time-series prediction. Aditya Grover et al. [64] proposed a deep learning-based approach for weather forecasting.

While deep learning-based wind prediction approaches have proved their capabilities in certain applications, they may limit their usefulness for real-time analyses due to high

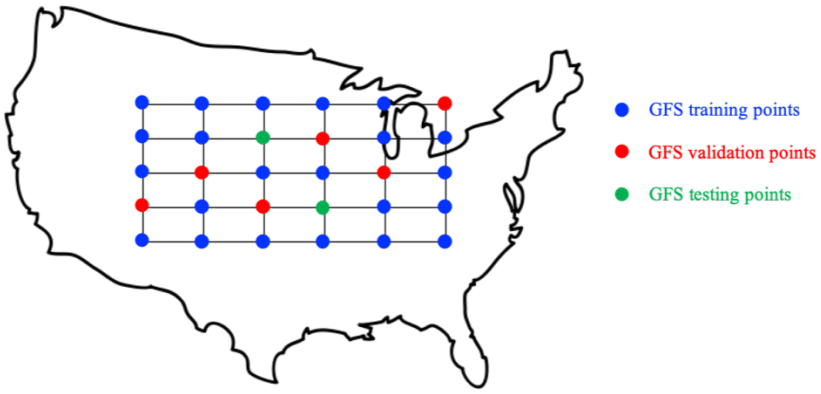
computational costs (e.g., constructing neural networks with numerous layers). On the other hand, supervised machine learning-based approaches, which are typically simpler than deep learning-based approaches, may produce predicted outcomes in an appropriate time frame. Given the aforementioned observations, the following research hypothesis can be developed:

> *Research Hypothesis 1.1: A supervised machine learning-based regression method will provide wind forecasts with a lower prediction error (i.e., RMSE) compared to a linear interpolation method.*

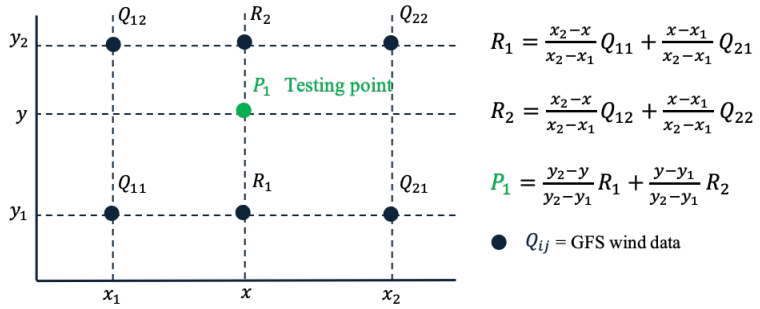
The Research Hypothesis 1.1 can be investigated and tested through the Research Experiment 1.1 (i.e., supervised machine learning-based regression vs. linear interpolation with respect to wind predictions) as notionally illustrated in Figure 4.2. More specifically, the first step is to decompose wind data (e.g., GFS eastward wind) into training, validation, and testing points. The testing points are randomly sampled in the U.S. territory. The second step is to predict wind information at the testing points by using a linear interpolation method. In a similar way, the third step is to forecast wind information at the testing points by utilizing a supervised machine learning-based regression method. Once two different predicted values at the testing points are obtained, the last step is to compare the values to the original data (e.g., GFS eastward or northward wind) and evaluate the two methods by calculating the Root Mean Square Error (RMSE).

> *Research Experiment 1.1: A supervised machine learning-based regression method is compared to a linear interpolation method in terms of testing points randomly sampled in the U.S. territory.*

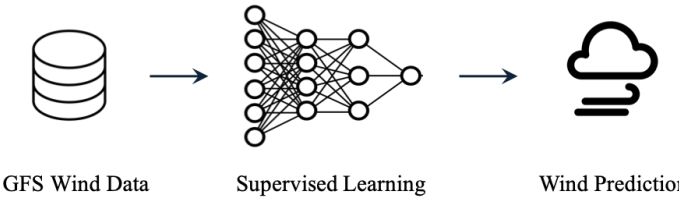
Additional details about the Research Experiment 1.1 will be discussed in section 5.2.



(a) Step 1. Decompose wind data into training, validation, and testing datasets



(b) Step 2. Predict winds at testing points using linear interpolation



(c) Step 3. Predict winds at testing points using supervised machine learning

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

n = Number of testing points
y_i = Actual wind
$\hat{y}_i$ = Predicted wind

(d) Step 4. Compare two approaches by calculating RMSEs

Figure 4.2: Steps for Research Experiment 1.1

4.2 Unsupervised Machine Learning-based Convective Weather Prediction

Convective weather is a significant factor that must be addressed in a real-time flight path optimization framework. There have been many efforts to create a model that accurately predicts convective weather activity. For example, James E. Evans and Elizabeth R. Ducot [65] from the Massachusetts Institute of Technology (MIT) Lincoln laboratory developed the Corridor Integrated Weather System (CIWS) that provides more accurate graphical areas of convective weather activity to help flight operators systematically control the congested U.S. airspace. Figure 4.3 shows an example visualization of the CIWS product generated by the MIT Consolidated Storm Prediction for Aviation (CoSPA) system [66].

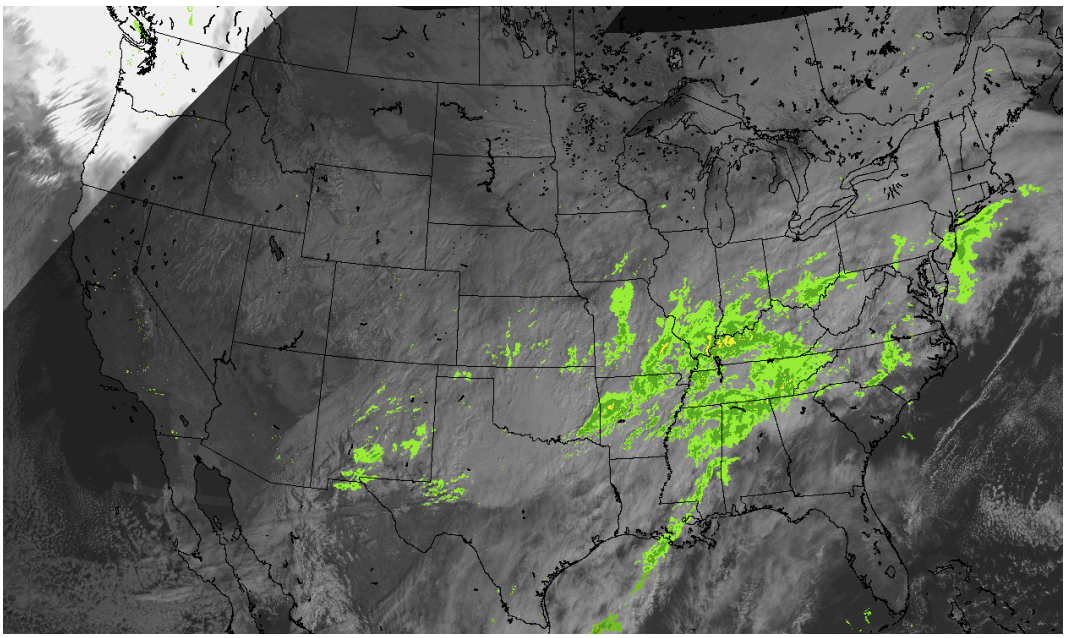


Figure 4.3: CIWS product visualization generated by the MIT CoSPA system

Although the CIWS was originally designed to automatically generate graphical depictions of the convective weather, the integration of convective weather data into a flight planning system was noted as a key challenge in the aviation industry. In order to transform the CIWS graphical areas into grid-based convective weather avoidance areas (e.g.,

polygon), Rich DeLaura et al. [67] developed the Convective Weather Avoidance Model (CWAM) by employing more than 500 aircraft-convective weather scenarios encountered in the Indianapolis airspace. The CWAM identified convective weather areas in which pilots were potentially guided to avoid the areas by investigating the planned and real flight trajectories in the areas [68]. Mikhail Rubnich and Rich DeLaura [69] further improved the effectiveness of the CWAM, which is now called the Convective Weather Avoidance Polygon (CWAP), as they noticed that the CWAM sometimes generated unrealistic avoidance areas. While the MIT Lincoln laboratory's convective weather model is a product with which NASA has experience in the context of several research projects, it is important to note that the product is not a commercially operational; thus, there is no guarantee of its availability in future years. The Weather Company (i.e., formerly WSI) also provides a convective weather product which has been coded as an in-house tool; however, this product is accessible to only customers [70].

The AWC publicly provides convective weather polygon information on a scheduled basis, which is widely used by many aviation researchers. In particular, the AWC uses radar and satellite data to define convective SIGMET polygon areas that are potentially hazardous to all aircraft as shown in Figure 4.4 [71]. It is important to note that the vertices that the AWC typically defines are most often five due to the limitation in the text version of the convective SIGMET data [72]. Although the AWC convective SIGMET product is currently utilized by many flight planning tools, the problem is that there are some potential limitations to the AWC convective weather data. For example, the convective SIGMET weather data is only updated hourly. This means that the convective SIGMET polygons do not change their shape over an hour despite the fact that the convective cells are moving and evolving. Moreover, the convective SIGMET weather data may contain inherent uncertainty because it consists of human-drawn polygons; thus, it is prone to human error. Therefore, a graphical representation of the AWC convective SIGMET data may not represent actual convective weather activity.

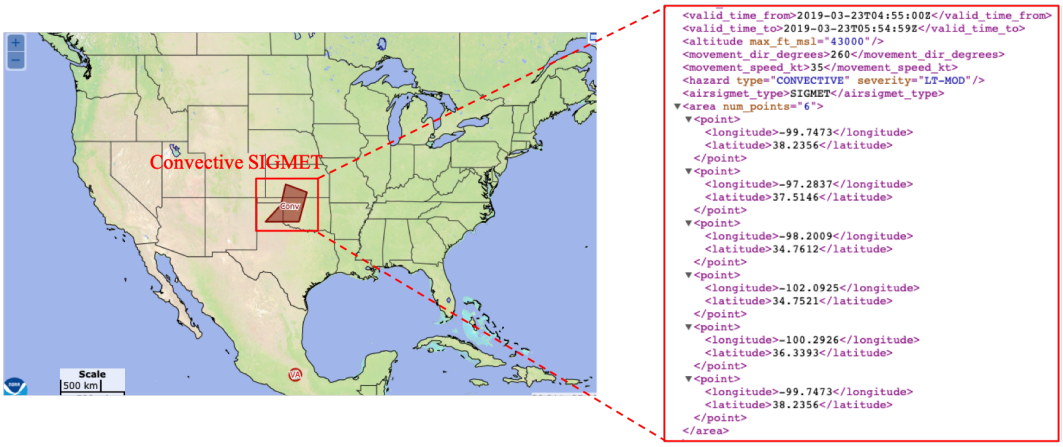


Figure 4.4: AWC convective SIGMET polygon at 2019-03-23 23:00 UTC

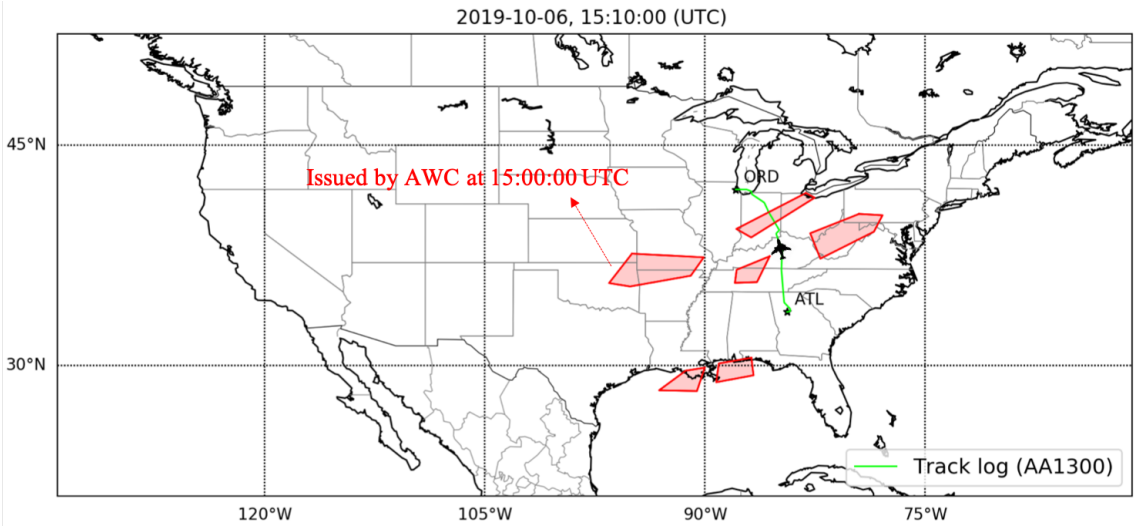


Figure 4.5: AA1300 flight path visualization with the AWC convective SIGMET data

For example, Figure 4.5 shows the trajectory of the previous American Airlines (AA) Flight 1300 from ATL to Chicago O'Hare International Airport (ORD) on October 6th, 2019. Given that aircraft is at the position (i.e., the black aircraft icon in Figure 4.5) at 15:10 Universal Time Coordinated (UTC), it indicates that the pilots actually penetrated one of the convective SIGMET polygons issued by the AWC at 15:00 UTC, which is unlikely to happen in reality because pilots are typically recommended to avoid hazardous weather areas. In fact, it is more likely that the pilots found a path clear of any convective weather

activity within the area covered by the AWC SIGMET data. Based on the aforementioned information, the following observation can be claimed:

> *Observation 1.2: The AWC convective weather data is widely used in the aviation industry; however, the dataset has some potential limitations in its operations; thus, a graphical representation of the AWC convective weather data does not always accurately reflect the actual convective weather activity.*

The Observation 1.2 is significant from the perspective of a computer simulation aiming to optimize flight routes under severe weather conditions. In the computer simulation environment, the graphical representation of the AWC convective SIGMET data is generally treated as a hard constraint; thus, it has to be avoided. This means that an algorithm may have to take an excessive deviation around hazardous weather activities if incorrect convective information is provided, implying that the simulated flight trajectory generated by the algorithm is not actually an optimal flight path in reality. On the other hand, once convective activities are modeled accurately in the computer simulation environment, the algorithm is able to tactically select more acceptable flight routes; thus, it can minimize operating costs in reality. Based on these observations, the following research question can be constructed:

> *Research Question 1.2: How can the areas of convective weather activity be defined more accurately than the AWC convective SIGMET polygon data?*

In fact, a primary question regarding the AA Flight 1300 case is, "Why did the pilots penetrate one of the AWC convective weather polygons in reality?" It could be hypothesized that some of the graphical area portions were not actually convective in reality. This hypothesis can be easily proven by retrieving all historical observational information such as ground-based weather radar data and the real flight trajectory information. Figure 4.6 provides an in-depth visualization with the AA 1300 flight trajectory and ground-based

weather radar data. As can be seen, there are three different color levels with respect to radar reflectivity (i.e., intensity of rainfall) values: 1) green, 2) yellow, and 3) red. Pilots are typically guided to avoid the red region (i.e., high radar reflectivity value area) by the regulation in the aviation industry. It appears that the pilots might seek to avoid the areas in which radar reflectivity values were approximately above 40 dBZ. It must be noted that the radar reflection does not have altitude information; thus, in reality, we may not really know whether or not pilots went into an area with a radar reflectivity exceeding 40 dBZ but we may claim that they found a path clear of any convective weather activity within the area according to the historical data.

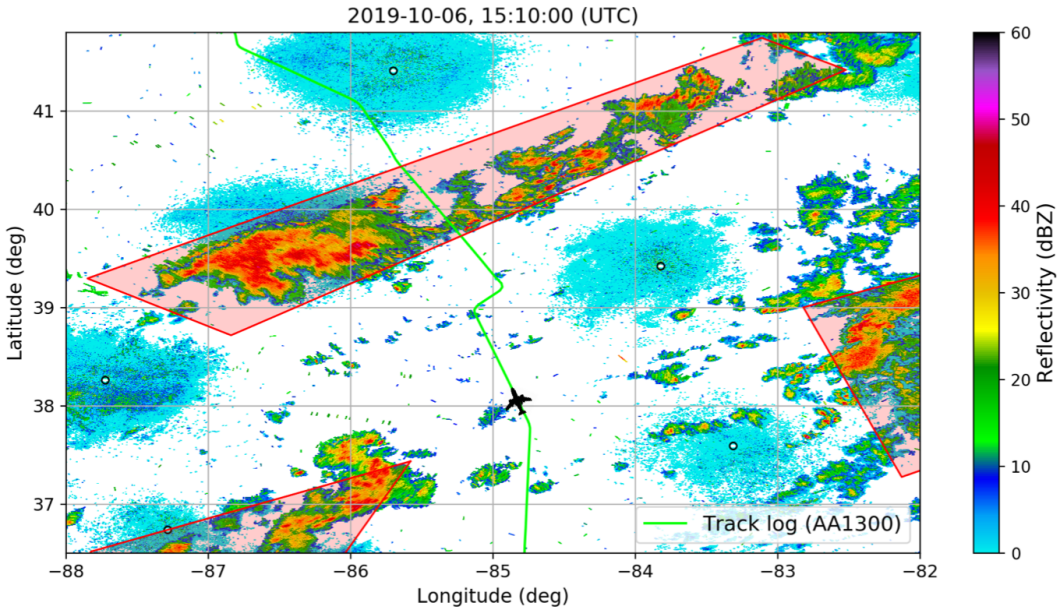


Figure 4.6: AA1300 flight path in-depth analysis with ground-based weather radar data

Given the aforementioned observations, it becomes obvious that ground-based observational information (e.g., radar data) is necessary to draw the boundaries of convective weather activity more accurately. However, it may still be challenging to come up with how to leverage multiple observational datasets to generate polygons representing current convective weather activities. One potential solution is to employ an unsupervised ma-

chine learning-based clustering algorithm because the ground-based observational data is typically provided as a scatter point on the U.S. territory. More specifically, if an algorithm is able to cluster the points, each cluster can generate a convex hull that potentially represents an area of hazardous weather activity. This implies that the AWC convective weather data, which is typically updated hourly, could be replaced by new polygons that are generated with multiple observational datasets which are updated every 10 minutes. Based on these observations, the following research hypothesis can be developed:

Research Hypothesis 1.2: An unsupervised machine learning-based clustering algorithm with multiple observational datasets which are updated every 10 minutes will define the areas of convective weather activity more accurately than the AWC convective weather data by generating convective weather polygons in a more frequent manner.

The Research Hypothesis 1.2 can be investigated and tested through the Research Experiment 1.2. More specifically, the first step is to collect all of the multiple observational datasets such as Next Generation Radar (NEXRAD) and Meteorological Aerodrome Report (METAR). The second step is to visualize polygon areas issued by the AWC. In a similar way, the third step is to visualize polygon areas that are generated by an unsupervised machine learning-based short-term (i.e., every 10 minutes) convective weather prediction method with the multiple observational datasets. The last step is to visually compare the AWC convective SIGMET data with new polygons generated by the proposed methodology.

Research Experiment 1.2: An unsupervised machine learning-based short-term convective weather prediction method is visually compared to the AWC convective SIGMET data.

Additional details about the Research Experiment 1.2 will be discussed in section 5.3.

4.3 Designated Points-based Flight Path Optimization

Path planning is one of the most important engineering problems, especially in air navigation systems. The key to solving the path planning problem is to identify the best pathfinding algorithm for a specific problem. Path planning problems have been solved by various types of approaches. In particular, there have been many attempts in the aviation industry to optimize flight paths by solving path planning problems.

The first type of approach is to utilize dynamic programming. For example, Hok K. Ng et al. [73] designed and evaluated the flight routing algorithm, which is based on dynamic programming, using the CWAM. Although the dynamic programming-based approach is capable of obtaining a global optimum solution without any local optimum concerns, the dynamic programming-based approach may suffer from high computational costs which may be unsuitable for real-time analyses. Another type of approach is to use local search algorithms. For instance, Luciano Blasi et al. [74] proposed a Particle Swarm Optimization (PSO)-based methodology for flight path generation. Christine Taylor and Craig Wanke [75] employed a Simulated Annealing (SA)-based algorithm to identify operationally acceptable flight routes impacted by severe weather. Robert A. Vivona et al. [76] presented a conflict resolution algorithm using the Genetic Algorithm (GA). In general, local search approaches are preferred to dynamic programming-based approaches due to the time constraint for real-time flight path planning applications. However, it is important to note that local search approaches do not always guarantee a global optimum.

After conducting a literature survey of flight path optimization [77, 78, 79, 80, 81, 82, 83, 84], a graph-based approach was identified as the most suitable flight path optimization algorithm for this research. The Dijkstra's algorithm, one of the popular graph-based approaches, guarantees the shortest path from the starting point to the goal if edges have only positive costs [85]. However, the Dijkstra's algorithm may take a very long time so the algorithm may not be applicable in a real-time flight path optimization framework. On

the other hand, the Greedy algorithm, which is another type of graph-based approaches, runs faster than the Dijkstra's algorithm but does not guarantee the shortest path solution. Figure 4.7 depicts how the algorithms find the shortest path differently.

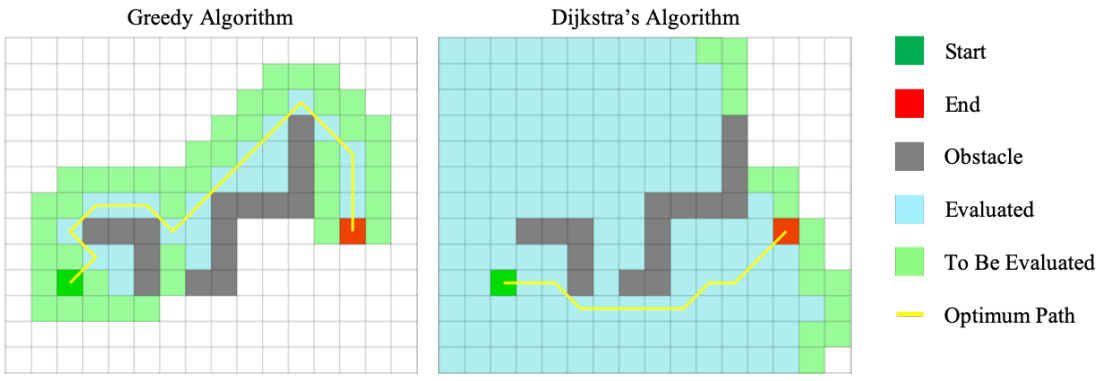


Figure 4.7: Visual comparison of the two pathfinding algorithms (Adapted from [86])

The A* search algorithm was introduced by combining the advantages (i.e., balancing between accuracy and speed) of the two previous algorithms [87]. Specifically, the A* search algorithm is a balance of the two previous methods with a combined cost function that accounts for the distance from start to the current point as well as from start to the goal. Although the A* search algorithm can suffer from high computational costs especially if it must account for the worst case, the algorithm is generally guaranteed to find an optimal flight path within fairly flexible computational costs (i.e., minimizing vertex expansion) [34]. Given that the A* search algorithm is employed to find an optimal flight route in a real-time flight path optimization framework, the obvious downside of the algorithm is that it finds an optimal flight path only with user-supplied information (e.g., pre-determined U.S. airspace). Based on the aforementioned information, the following observation can be claimed:

Observation 2.1: The A search algorithm is the most appropriate pathfinding approach for a real-time flight path optimization framework; however, the A* search algorithm finds an optimal flight path only with user-supplied information.*

The Observation 2.1 is significant in reality because the framework that uses the A* search algorithm may be constrained to the pre-determined U.S. airspace infrastructure (e.g., FAA pre-determined airway), potentially leading to not being able to provide more flight route options that can occur in reality. In fact, pilots sometimes do not follow established waypoint-to-waypoint air routes but choose other flight route options especially when they encounter hazardous weather activity. In the current ATC operation, it is common to set a certain lateral deviation to avoid such a weather event instead of directing established waypoint-to-waypoint airways. For example, Figure 4.8 shows the trajectory of the previous Delta Airlines 2143 Flight on August 3rd, 2019, indicating that the pilots did not follow established waypoint-to-waypoint air routes (i.e., black dashed lines). Based on these observations, the following research question can be constructed:

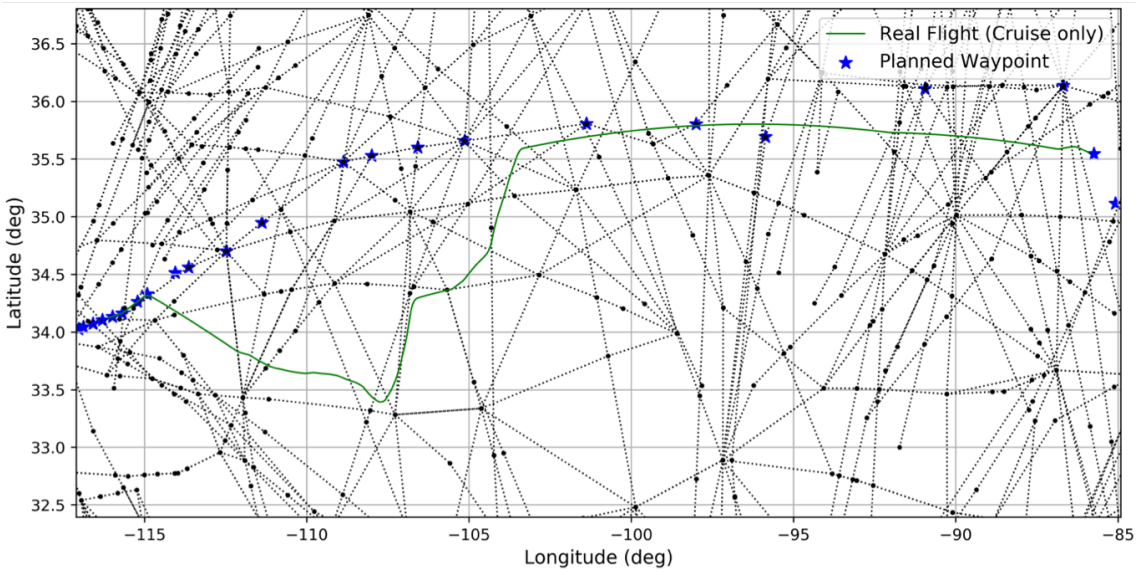


Figure 4.8: DL2143 flight path visualization (en-route only) with the planned waypoints

Research Question 2.1: How can the constraint of the A search algorithm (i.e., find an optimal flight path using only pre-determined airways) be relaxed to provide more flight route options that are available to pilots?*

To provide more flight route options that potentially relax the pre-determined U.S. airspace infrastructure constraint, the concept of free flight has been introduced for future air transportation systems. For example, Xuxi Yang et al. [88] designed a real-time collision avoidance algorithm using the Monte Carlo Tree Search (MCTS) method as one of the enablers for free-flight operations. While a free-flight approach can provide more freedom in a flight path selection, it must be noted that the free-flight approach may be trapped in a local optimum. More importantly, even though the current ATC operation does not strictly ask pilots to comply with the ATC requirements (e.g., allowing them to make a certain lateral deviation to avoid hazardous weather conditions), the free-flight approach may not be able to model the situation where aircraft must follow established waypoint-to-waypoint flight routes that are generated based on existing waypoints. Given the aforementioned observations, the following research hypothesis can be developed:

Research Hypothesis 2.1: A hybrid method that combines the A search algorithm with a free-flight approach will automatically provide flight route opportunities that are not necessarily constrained to established waypoint-to-waypoint airways but rather generate acceptable free-flight route options.*

The Research Hypothesis 2.1 can be investigated and tested through the Research Experiment 2.1. More specifically, the first step is to select the date where hazardous convective weather activities (e.g., hurricane Laura) are dominant and collect corresponding historical weather data for the selected date. Under the weather conditions, the second step is to generate two optimal flight routes with different options: 1) use a designated points-based flight path optimization model (i.e., hybrid method combining the A* search

algorithm with a free-flight approach), which is proposed in this research, and 2) find an optimal flight route using only the A* search algorithm. The last step is to compute the Hausdorff distance for the two scenarios to compare them with a real flight trajectory created by flight dispatchers of major airlines. Here, the Hausdorff distance is defined as a metric that measures the degree of mismatch between two subsets [89]. One way to understand the Hausdorff distance in the Research Experiment 2.1 is to think of it as a metric that determines how a flight trajectory generated by simulation is similar to a real flight trajectory. Intuitively, the smaller the Hausdorff distance is, the more similar the two trajectories are.

Research Experiment 2.1: The Hausdorff distances for simulated optimum flight routes generated by both a designated points-based flight path optimization model (i.e., RTOP-v3) and the A search algorithm (i.e., RTOP-v2) are computed to see which route is more similar to a real flight trajectory, especially under severe weather conditions.*

Additional details about the Research Experiment 2.1 will be discussed in section 5.4.

4.4 Summary of Research Statements

The research formulation presented in Chapter 4 reflected a scientific method. The scientific method generally guides the following steps as shown in Figure 4.9: 1) motivate a problem, 2) establish a research objective, 3) conduct literature search and identify a research gap, 4) define a research question, 5) form a research hypothesis, 6) test the research hypothesis through a research experiment, 7) interpret a result, and 8) draw a conclusion.

This section is particularly intended to summarize the research statement formulated for this dissertation. Chapter 5 will present the implementation of the proposed methodology. Chapter 6 interprets a series of results obtained through a set of case studies and statistical analyses. Finally, Chapter 7 draws a conclusion for this dissertation.

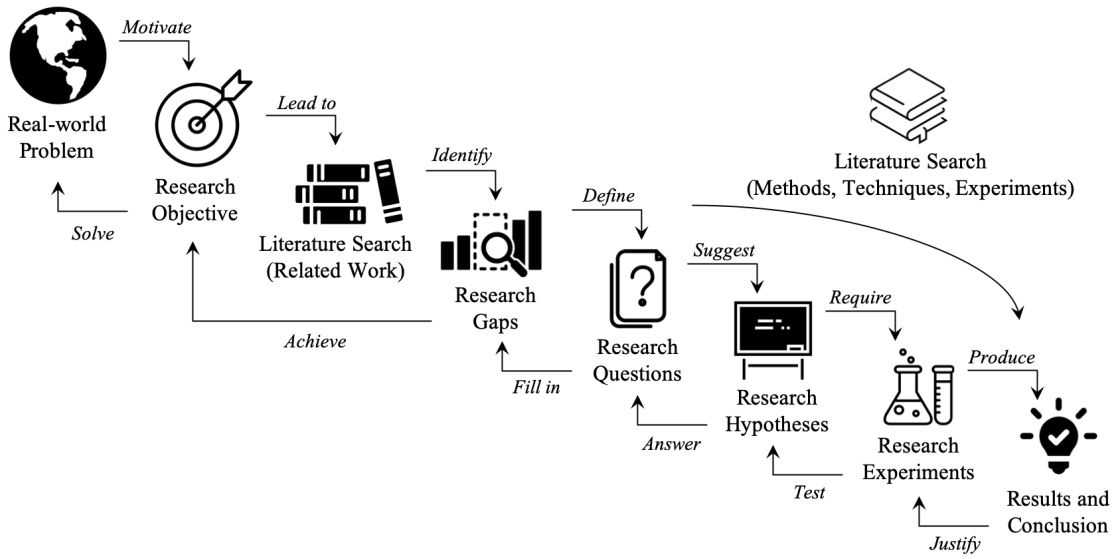


Figure 4.9: Decomposition and recomposition of the interpretation of the ASDL Ph.D. process (Adapted from [90])

Step 1. Motivate a problem

This research starts by observing two potential issues related to current in-flight re-planning systems identified from the interviews with the pilots. The identified issues are as follows: First, current in-flight re-planning systems are not fully automated; thus, pilots today perform some portions of the in-flight activities manually. Second, weather forecasts used for current in-flight re-planning systems are not always accessible in a timely manner.

Step 2. Establish a research objective

This research attempts to resolve aforementioned potential issues by developing a framework that automatically performs in-flight re-planning continuously with the latest weather information sets available. The intent of this research is to provide the automated framework with two use-cases in mind: 1) to help flight dispatchers at major airlines by providing the capability to continuously re-route flights using the latest weather information and 2) to alleviate the cockpit workload of pilots employed by small airline operators that do not necessarily have flight dispatchers but ask the pilots to generate and update flight plans.

Step 3. Conduct literature search and identify a research gap

In relation to the research objective, the literature review outlines several attempts to develop capabilities improving current in-flight re-planning systems and identifies research gaps under the following areas: 1) flight management system, 2) ground-based flight planning framework, 3) electronic flight bag, and 4) machine learning-based flight route prediction. The literature search results in the following research gaps:

- Research Gap 1: A flight management system reduces the workload of the flight crew by automating a variety of in-flight activities but some parts in the flight planning functionality are still manual input.
- Research Gap 2: A ground-based flight planning framework has mostly automated the task of flight planning for flight dispatchers but the framework does not address the connectivity issue of co-constructing knowledge between pilots and flight dispatchers, leading to disapproved requests resulting in inefficient communication.
- Research Gap 3: The most widely used aviation-related applications implemented in the electronic flight bag have proved their capabilities in certain application areas; however, the applications may not be feasible for a real-time flight path optimization framework that uses the latest weather information to continuously update flight routes.
- Research Gap 4: Deep learning techniques are prevalently adopted in the aviation industry; however, these techniques may not be feasible for a real-time flight path optimization framework due to high computational costs.

Step 4. Define a research question

In an effort to fill in the research gaps, this research defines an overarching research question as follows:

- Overarching Research Question: How can a cockpit-based real-time flight path optimization framework, which automatically performs in-flight re-planning continuously with the latest weather information, be developed?

The overarching research question poses three different research problems (i.e., supervised machine learning-based wind regression, unsupervised machine learning-based convective weather prediction, and designated points-based flight path optimization) that are specifically formulated to answer research questions as follows:

- Research Question 1: How can the capability of providing weather information accurately and continuously to a cockpit-based real-time flight path optimization framework within a specified time be developed?
- Research Question 2: How can the capability of providing simulated optimum flight routes automatically to a cockpit-based real-time flight path optimization framework be developed?

The research questions (i.e., RQ1 and RQ2) naturally lead to conduct the literature search, resulting in the following observations:

- Observation 1.1: The most common approach for obtaining continuous wind information in current in-flight re-planning systems is via a linear interpolation method; however, the linear interpolation method may have some limitations in predicting wind patterns where non-linear behavior is dominant.
- Observation 1.2: The AWC convective weather data is widely used in the aviation industry; however, the dataset has some limitations in its operations; thus, a graphical representation of the AWC convective weather data does not always accurately reflect the actual convective weather activity.
- Observation 2.1: The A* search algorithm is the most appropriate pathfinding approach for a real-time flight path optimization framework; however, the A* search algorithm finds an optimal flight path only with user-supplied information.

Based on these observations, the following research questions are developed:

- Research Question 1.1: How can continuous wind information be provided more accurately (i.e., lower prediction error) other than using the current linear interpolation method?
- Research Question 1.2: How can the areas of convective weather activity be defined more accurately than the AWC convective SIGMET polygon data?
- Research Question 2.1: How can the constraint of the A* search algorithm (i.e., find an optimal flight path using only pre-determined airways) be relaxed to provide more flight route options that are available to pilots?

Step 5. Form a research hypothesis

The research questions (i.e., RQ1.1, RQ1.2, and RQ2.1) naturally lead to construct research hypotheses as follows:

- Research Hypothesis 1.1: A supervised machine learning-based regression method will provide wind forecasts with a lower prediction error (i.e., RMSE) compared to a linear interpolation method.
- Research Hypothesis 1.2: An unsupervised machine learning-based clustering algorithm with multiple observational datasets which are updated every 10 minutes will define the areas of convective weather activity more accurately than the AWC convective weather data by generating convective weather polygons in a more frequent manner.
- Research Hypothesis 2.1: A hybrid method that combines the A* search algorithm with a free-flight approach will automatically provide flight route opportunities that are not necessarily constrained to established waypoint-to-waypoint airways but rather generate acceptable free-flight route options.

Step 6. Test the research hypothesis through a research experiment

The research hypotheses are investigated and tested through the following research experiments:

- Research Experiment 1.1: A supervised machine learning-based regression method is compared to a linear interpolation method in terms of testing points randomly sampled in the U.S. territory.
- Research Experiment 1.2: An unsupervised machine learning-based short-term convective weather prediction method is visually compared to the AWC convective SIGMET data.
- Research Experiment 2.1: The Hausdorff distances for simulated optimum flight routes generated by both a designated points-based flight path optimization model (i.e., RTOP-v3) and the A* search algorithm (i.e., RTOP-v2) are computed to see which route is more similar to a real flight trajectory, especially under severe weather conditions.

Ultimately, this research proposes a data-driven approach that leverages various machine learning and optimization algorithms to answer the research questions, leading to achieve the research objective. The proposed methodology is demonstrated with a set of case studies and statistical analyses to prove the applicability and potential benefits of the framework developed in this research. Additional details about the implementation of the proposed methodology will be discussed in Chapter 5.

CHAPTER 5

IMPLEMENTATION

This research constructed three different research hypotheses to demonstrate two requirements raised by the two research questions: 1) the capability of providing weather information accurately and continuously and 2) the capability of providing simulated optimum flight routes automatically. The research hypotheses presented in Chapter 4 are as follows:

- Research Hypothesis 1.1: A supervised machine learning-based regression method will provide wind forecasts with a lower prediction error (i.e., RMSE) compared to a linear interpolation method.
- Research Hypothesis 1.2: An unsupervised machine learning-based clustering algorithm with multiple observational datasets which are updated every 10 minutes will define the areas of convective weather activity more accurately than the AWC convective weather data by generating convective weather polygons in a more frequent manner.
- Research Hypothesis 2.1: A hybrid method that combines the A* search algorithm with a free-flight approach will automatically provide flight route opportunities that are not necessarily constrained to established waypoint-to-waypoint airways but rather generate acceptable free-flight route options.

Given that the two requirements (i.e., capabilities of continuously providing the latest weather information accurately and simulated optimum flight routes automatically) are satisfied by demonstrating the aforementioned research hypotheses, this research established the Overall Research Hypothesis as follows:

Overall Research Hypothesis: If a cockpit-based real-time flight path optimization framework developed in this research utilizes 1) accurate wind field datasets predicted by a supervised machine learning-based wind model (i.e., Enabler 1), 2) reliable convective weather polygon areas generated by an unsupervised machine learning-based convective weather model (i.e., Enabler 2), and 3) flight planning functionalities by a designated points-based flight path optimization model (i.e., Enabler 3), then the framework can automatically provide simulated flight route options shorter than real-world flight routes by continuously optimizing the simulated flight trajectories.

To substantiate the Overall Research Hypothesis formulated for this dissertation, this research proposes a data-driven approach that leverages various machine learning and optimization algorithms. Figure 5.1 shows a research roadmap used for this dissertation.

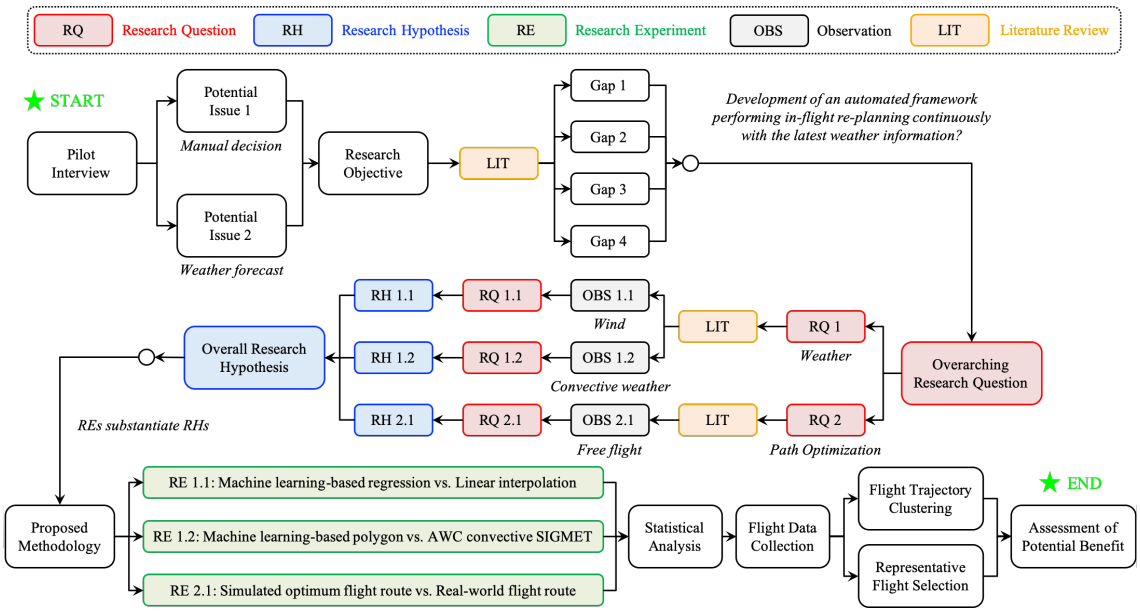


Figure 5.1: Research roadmap

The proposed methodology, which is called the RTOP-v3, implements three different

steps that include machine learning, data integration, and flight path optimization as shown in Figure 5.2. More specifically, in the machine learning step, the RTOP-v3 employs Enabler 1 (i.e., supervised machine learning-based wind model) to obtain continuous wind information in the U.S. territory and Enabler 2 (i.e., unsupervised machine learning-based convective weather model) to forecast reliable convective weather activity. Once weather prediction datasets are fed into the framework developed in this research, in the data integration step, all of the necessary datasets such as FAA pre-determined U.S. airspace infrastructure are integrated for the next step (i.e., flight path optimization). In the flight path optimization step, the RTOP-v3 employs Enabler 3 (i.e., designated points-based flight path optimization model) to optimize flight routes.

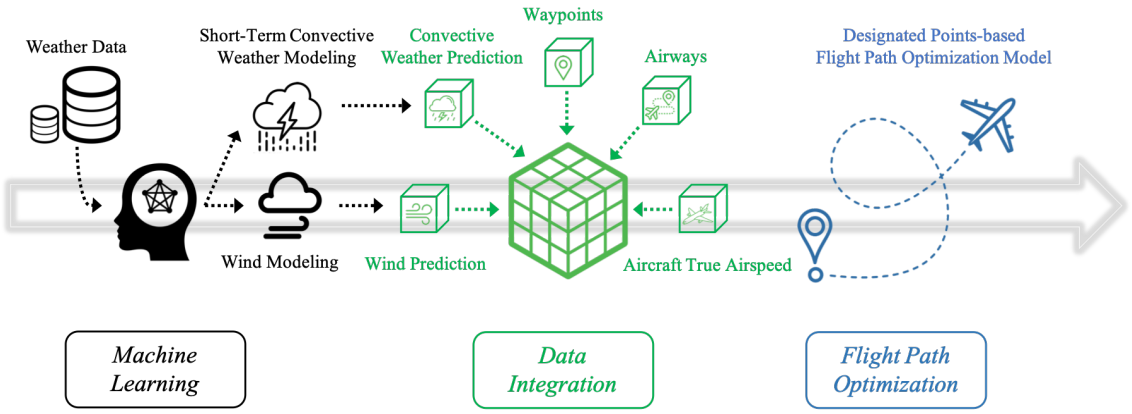


Figure 5.2: Research overview

The remainder of this chapter is organized as follows. The first section describes all of the datasets (e.g., wind, radar, and waypoint) used for the proposed methodology. The second section introduces the three different Enablers (i.e., supervised machine learning-based wind model, unsupervised machine learning-based convective model, and designated points-based flight path optimization model) that are specifically designed to substantiate the research hypotheses (i.e., RH 1.1, RH 1.2, and RH 2.1) presented in Chapter 4. The last section illustrates the software architecture developed for this research.

5.1 Data Preparation

5.1.1 Global Forecast System (GFS)

After careful consideration of the most well-known numerical weather models summarized in Table 4.1, the GFS model is selected as a primary wind dataset for this research because 1) the GFS model is commercially available so that users can easily connect to the server and download datasets, 2) the GFS model covers the entire globe in addition to the U.S. territory, which potentially shows a possibility in which this research can be extended to an international flight analysis, and 3) the NOAA recently made several significant technological improvements on the GFS model [91]. The GFS model is a global weather forecast model produced by the National Centers for Environmental Prediction (NCEP) and can be downloaded from the official website [92]. The GFS model is updated every six hours (i.e., 00, 06, 12, 18 UTC) and provides a set of detailed weather-related properties such as temperature, pressure, wind speed/direction, and relative humidity against longitude, latitude, altitude, and timestamp. The GFS model has an approximate horizontal resolution of 13 kilometers and divides the Earth into 64 layers vertically.

A data pipeline (i.e., *WIND.py*) is implemented to retrieve the GFS weather data from the NOAA data server and to automatically go through data pre-processing steps. Among various GFS weather parameters (e.g., temperature), this research primarily concentrates on wind speed and direction datasets. The data pre-processing steps implemented in the data pipeline are as follows: 1) decode an original GFS weather data using the Pygrib [93] library, 2) reconstruct wind datasets from a two-dimensional table to an input-output column for a supervised machine learning process, and 3) decompose the wind datasets into training (80%) and validation (20%) datasets. Here, the validation datasets are randomly selected. Figure 5.3 shows an example visualization of the GFS wind field datasets at a specific time and altitude generated by the framework developed in this research.

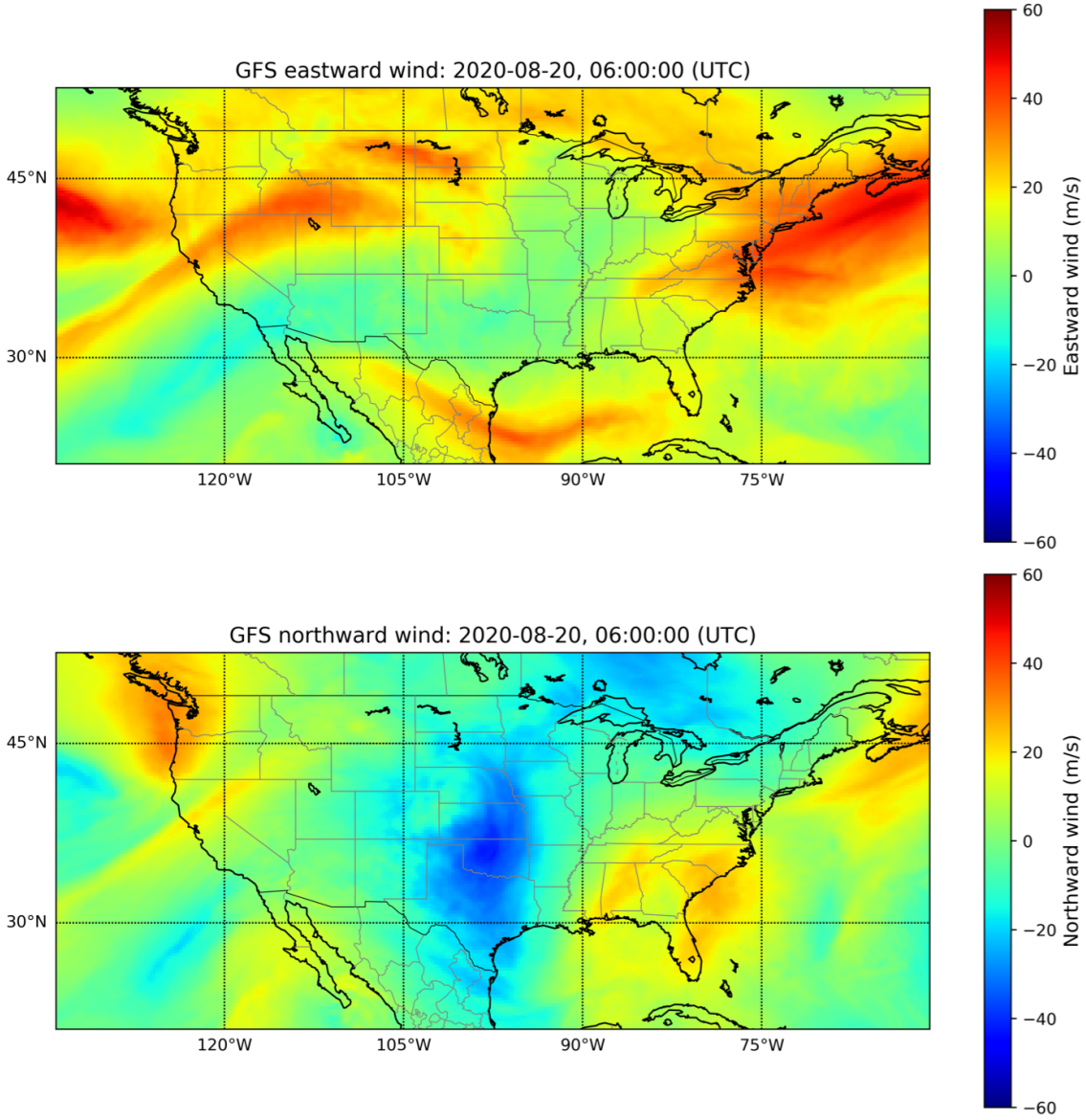


Figure 5.3: GFS wind visualization at 2020-08-20 06:00 UTC (Altitude = 250 hPa)

5.1.2 Next Generation Radar (NEXRAD)

The NEXRAD system in the U.S. currently consists of 159 radar sites in which the Weather Surveillance Radar 1988 Doppler (WSR-88D) technology is implemented as shown in Figure 5.4 [94]. The radar sites have an automated system that transmits precipitation information by sweeping around 360 degrees. Each scan typically takes five minutes to complete and it is then broadcast over the NOAA satellite network, which takes approximately 10 minutes in total. More specific information regarding the radar sites is found on Appendix C.

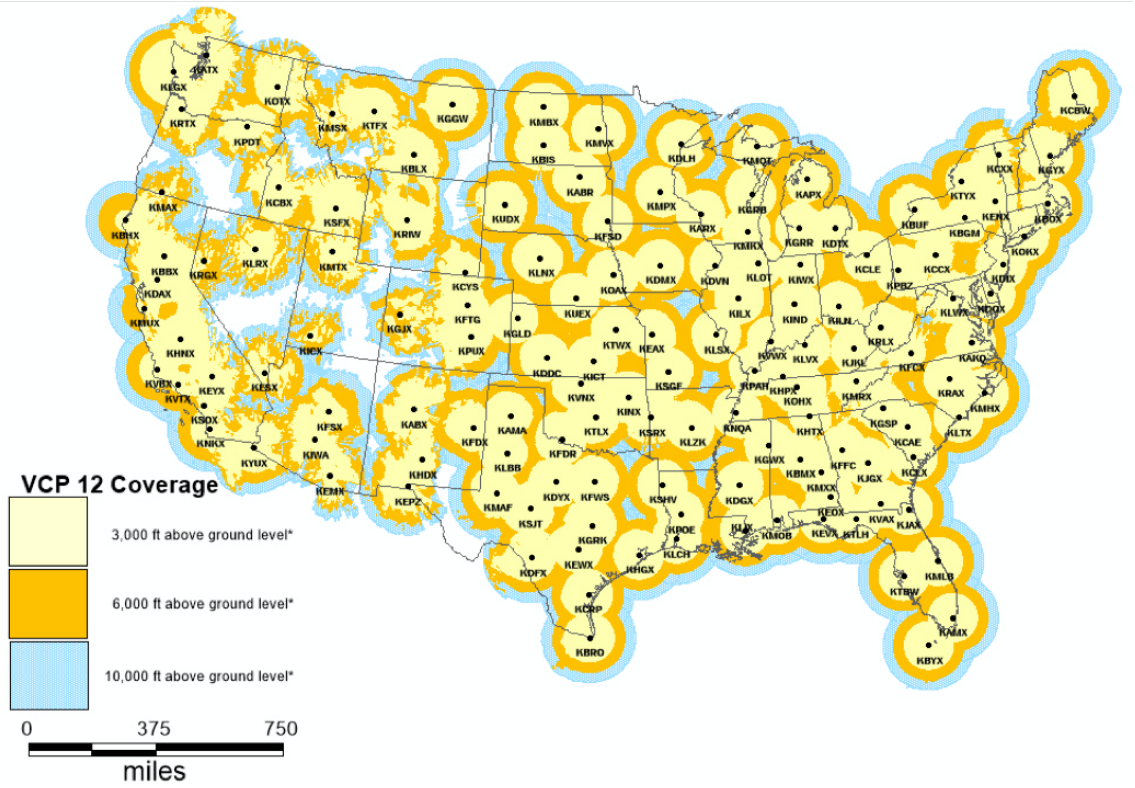


Figure 5.4: NEXRAD coverage visualization

Based on the fact that forecasters primarily use precipitation information for generating convective SIGMET polygons [72], a Python code (i.e., *SIGMET.py*) is developed to automatically go through data pre-processing steps on radar reflectivity information. Figure 5.5

shows an example visualization of the selected NEXRAD site locations and radar reflectivity values at a specific time, which was generated by the framework developed in this research. It is worth mentioning that the scale of radar reflectivity (dBZ) values is directly associated with the intensity of rainfall. In other words, a high radar reflectivity value fundamentally represents a high intensity of rainfall. For example, Table 5.1 [95] shows the relationship between radar reflectivity values and the intensity of rainfall.

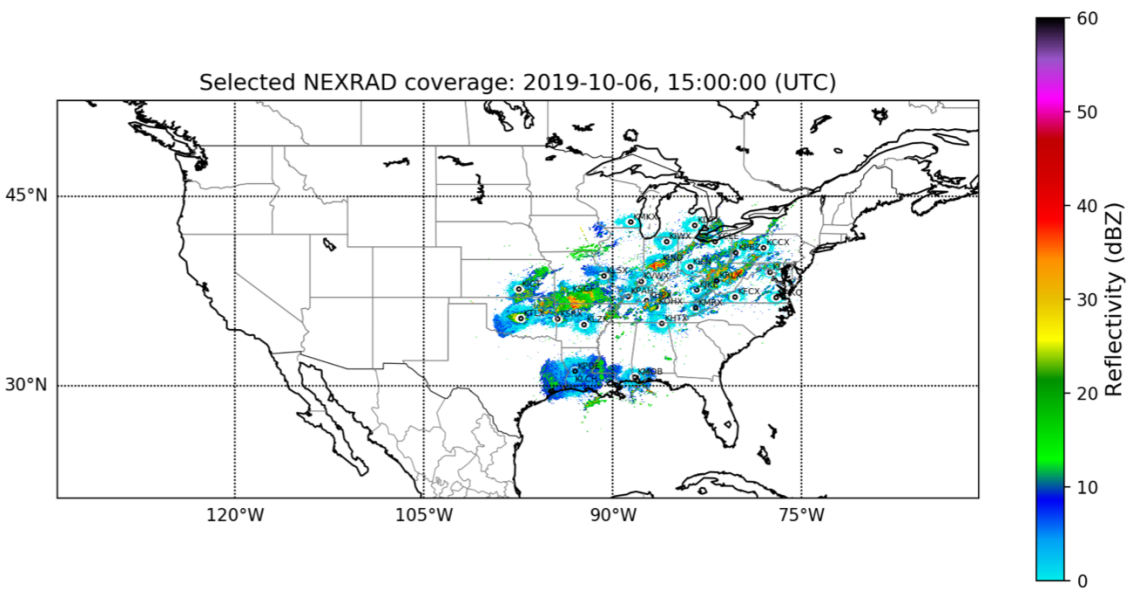


Figure 5.5: Selected NEXRAD visualization at 2019-10-06 15:00 UTC

Table 5.1: Relationship between radar reflectivity and intensity of rainfall

Reflectivity (dBZ)	Intensity
Less than 30 dBZ	Light
Between 30 and 40 dBZ	Moderate
Between 40 and 50 dBZ	Heavy
Greater than 50 dBZ	Extreme

5.1.3 Convective Significant Meteorological Information (SIGMET)

The AWC publicly issues weather alerts in the form of either an Airmen's Meteorological Information (AIRMET), non-convective SIGMET, or convective SIGMET data for different weather-impacted reasons. This research concentrated only on collecting convective SIGMET data, which represent polygon areas potentially hazardous to aircraft, because convective weather activity typically causes a majority of weather-related flight delays. The AWC convective SIGMET data is issued hourly on a scheduled basis and the shapes are valid up to two hours into the future and includes information with respect to initial position and velocity. It is important to note that the polygon shapes are forecasted based on a highly trained meteorologist's assessment of the hazard and typically defined with five vertices due to limitations in the text version of the SIGMET data.

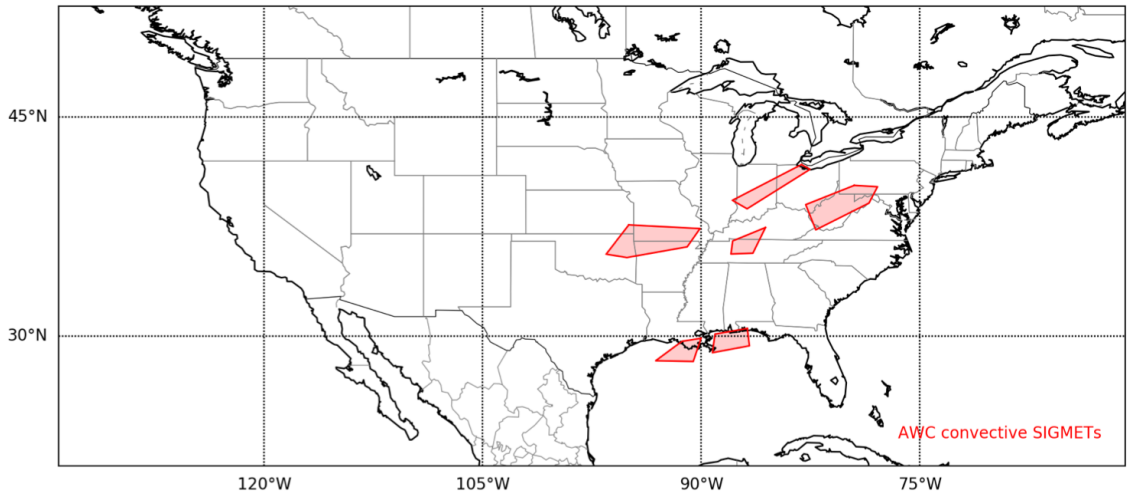


Figure 5.6: AWC convective SIGMET data visualization at 2019-10-06 15:00 UTC

A Python code (i.e., *DATA.py*) is developed to automatically connect the AWC Text Data Server (TDS) at regular intervals because the AWC has limited access to historical data. Figure 5.6 shows an example visualization of the convective SIGMET data at a specific time, which was generated by the framework developed in this research.

5.1.4 Meteorological Aerodrome Report (METAR)

Weather stations in the U.S. report weather-related information such as wind speed and direction every hour in a form of the METAR. The METAR report typically includes the following information [95]: 1) report type, 2) station, 3) date and time, 4) wind, 5) visibility, 6) weather phenomena, 7) sky condition, 8) temperature, and 9) dew point. For example, Figure 5.7 shows a sample of the METAR symbols.

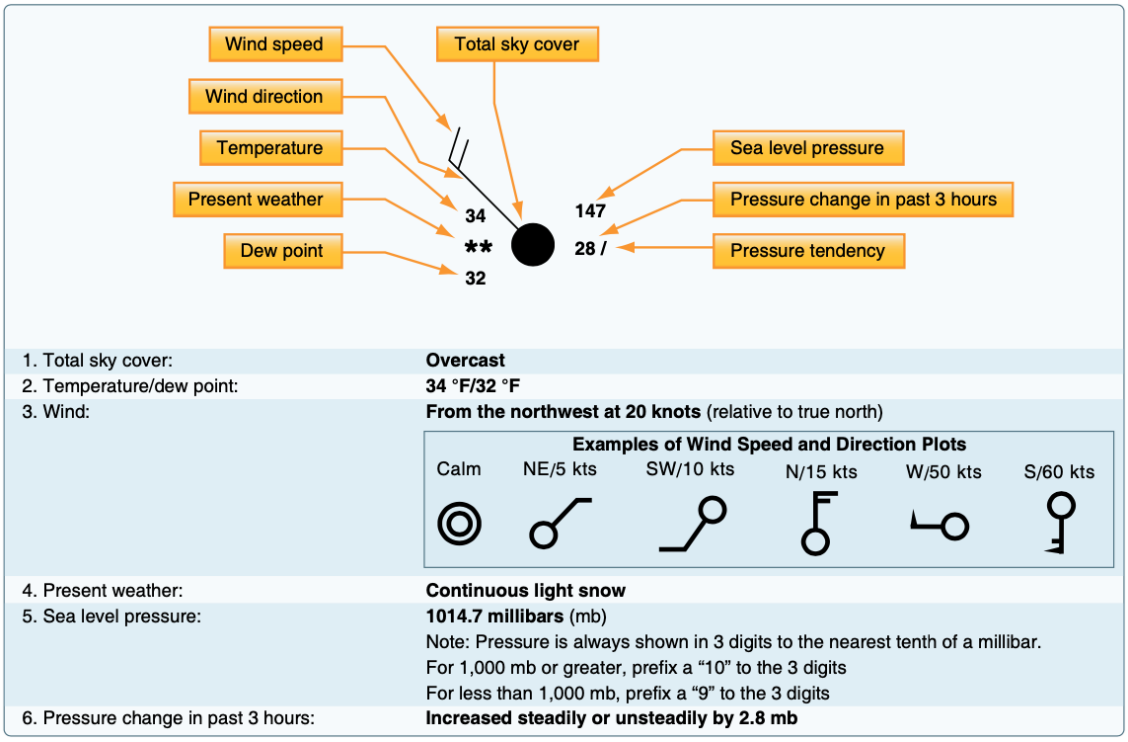


Figure 5.7: METAR symbol sample (Reprinted from [95])

Although METARs cover an area surrounding the immediate vicinity of an airport, a Python code (i.e., *SIGMET.py*) is developed to automatically download and decode the text version of the METARs because viewing many METARs together may provide a better picture of convective weather activity over the U.S. territory. Figure 5.8 shows an example visualization of the wind-related METAR information at a specific time and location. The

weather phenomena information TS, RA, and BR in Figure 5.8 stand for Thunderstorm, Rain, and Mist respectively.

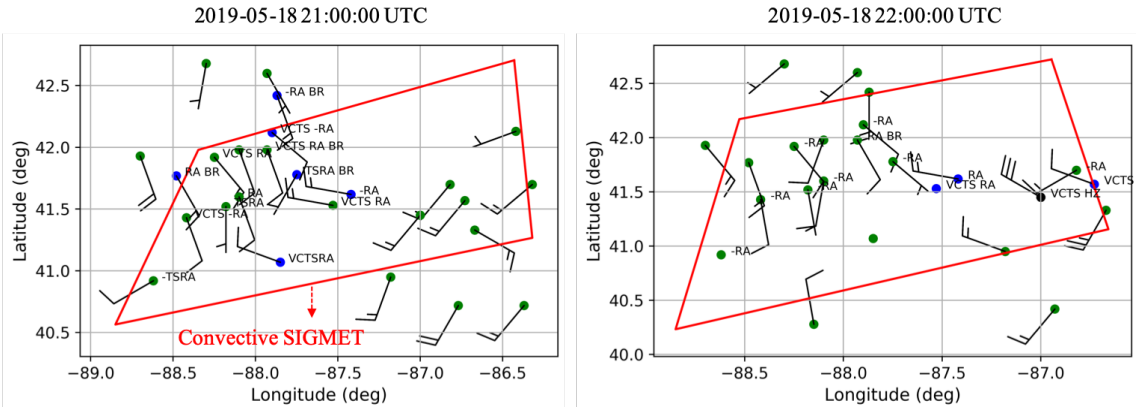


Figure 5.8: Wind-related METAR visualization at 2019-05-18 21:00 and 22:00 UTC

5.1.5 Pilot Report (PIREP)

The Pilot Report (PIREP) is a report by a pilot and can contain information about hazardous weather conditions (e.g., icing and turbulence). It is not mandatory for pilots to create PIREPs when they encounter unexpected weather conditions; however, pilots are encouraged to create a report because it provides valuable information regarding real weather conditions. Once pilots create a PIREP, it is typically transmitted in almost real-time to a ground station. While the convective SIGMET polygon areas are useful given that they typically cover relatively large geographic areas, the PIREP may provide more accurate information of the current convective weather activity compared to the blanket guidance generally provided by the AWC convective SIGMET data. For this reason, a Python code (i.e., *SIGMET.py*) is developed to automatically download and decode the text version of PIREP. Figure 5.9 shows an example the PIREP with the raw text data at a specific time.

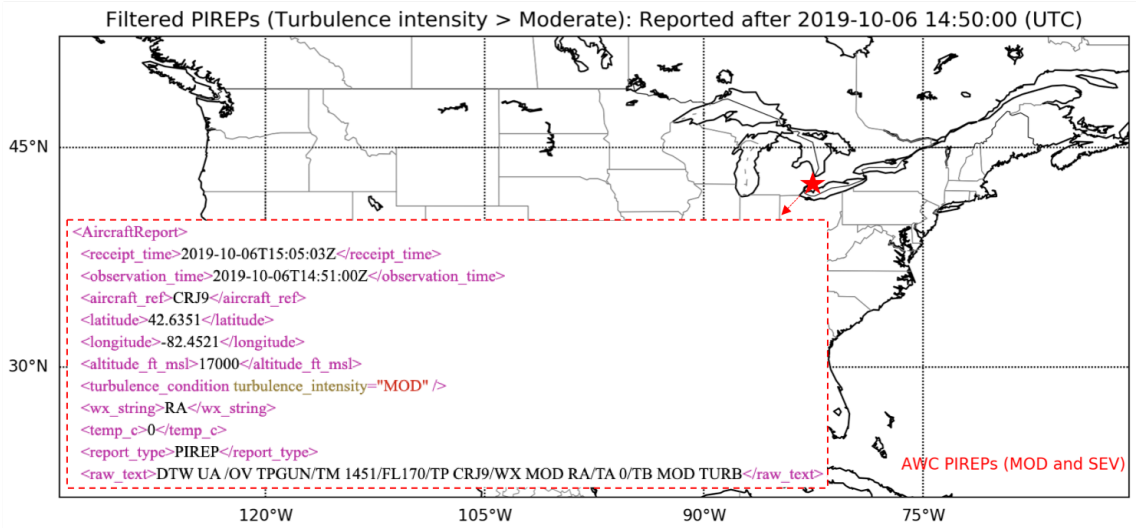


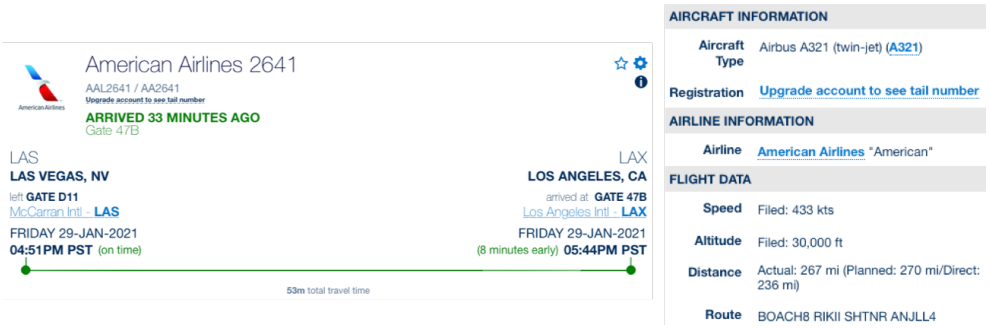
Figure 5.9: AWC PIREP visualization with the raw text data at 2019-10-06 14:50 UTC

5.1.6 Flight Tracking Data

FlightAware is an aviation company that provides flight tracking data platform service by receiving data from the FlightAware's network of ADS-B ground stations in many countries [96]. The company provides all tracking information (i.e., flight track log) that contains timestamp, ground speed, altitude, latitude, longitude, and course direction with the origin, destination, airline, flight number, and operating aircraft. For example, Figure 5.10 shows aircraft information, airline information, planned route information, and flight tracking information about the previous American Airlines Flight 2641 provided by FlightAware. This research specifically concentrates on collecting all tracking and true airspeed information for several flights, which came from the flight plan data called Air Navigation Service Provider (ANSP), for a verification and validation purpose.

5.1.7 Air Traffic Service (ATS) Route and Waypoint

A designated point is defined as a geographical position (e.g., longitude and latitude) that is pre-determined by the FAA in U.S. airspace. An Air Traffic Service (ATS) route is a specified route designed to systematically and efficiently manage air traffic in U.S. airspace. The



(a) Aircraft and airline information

KLAS BOACH8 RIKII SHTNR ANJLL4 KLAX							
Name	Latitude	Longitude	Outbound Course	Distance this Leg	Distance Remaining	Distance Flown	Type
KLAS	36.0800439	-115.1522347	284°	n/a	236	0	Origin Airport
BESSY	36.1076944	-115.2896111	201°	8	232	8	Waypoint
WITLA	35.9783056	-115.3508333	202°	10	223	13	Waypoint
JEBBB	35.9106389	-115.3851389	162°	5	219	18	Waypoint
BOACH	35.6782222	-115.2946944	182°	17	214	29	Reporting Point
RIKII	35.5058056	-115.3015000	211°	12	207	41	Waypoint
SHTNR	35.1246944	-115.5778333	226°	31	181	70	Waypoint
SMASH	34.7743333	-116.0238611	226°	35	148	103	Waypoint
SALYY	34.4168611	-116.4770278	244°	36	115	137	Waypoint
GLESN	34.3347222	-116.6763889	244°	13	103	148	Waypoint
ANJLL	34.2111944	-116.9827778	252°	19	84	166	Waypoint
CAANN	34.1658056	-117.1505833	252°	10	74	174	Waypoint
BOYEL	34.1053056	-117.3702500	253°	13	61	185	Waypoint
CRCUS	34.0731944	-117.4984444	260°	8	53	192	Waypoint
KLAX	33.9424964	-118.4080486	n/a	53	0	236	Destination Airport

(b) Planned route information

Time (EST)	Latitude	Longitude	Course	kts	feet	Rate	Reporting Facility
Fri 07:51:00 PM	**Left Gate (LAS) @ Friday 04:51:00 PM PST**						Airline
Taxi Time: 9 minutes							
Fri 08:00:32 PM	**Departure (LAS) @ Friday 05:00:32 PM PST**						FlightAware ADS-B (KLAS)
Fri 08:00:32 PM	36.0765	-115.1462	← 270°	149	2,550		FlightAware ADS-B (KLAS)
Fri 08:00:48 PM	36.0767	-115.1587	← 272°	141	3,400	2,766 ↑	FlightAware ADS-B (KVGT)
Fri 08:01:04 PM	36.0770	-115.1723	← 271°	150	4,025	2,156 ↑	FlightAware ADS-B (KVGT)
Fri 08:01:20 PM	36.0770	-115.1860	← 269°	164	4,550	1,781 ↑	FlightAware ADS-B (KLAS)
Fri 08:01:36 PM	36.0769	-115.2019	← 270°	185	4,975	1,828 ↑	FlightAware ADS-B (KLAS)
Fri 08:01:52 PM	36.0767	-115.2204	← 268°	209	5,525	2,156 ↑	FlightAware ADS-B (KLAS)
Fri 08:02:08 PM	36.0728	-115.2411	← 247°	232	6,125	2,143 ↑	FlightAware ADS-B (KHND)
Fri 08:02:27 PM	36.0586	-115.2605	✓ 217°	252	6,775	1,543 ↑	FlightAware ADS-B (KLAS)
Fri 08:02:43 PM	36.0403	-115.2698	↓ 199°	259	7,025	522 ↑	FlightAware ADS-B (KHND)
Fri 08:03:13 PM	36.0075	-115.2844	↓ 200°	267	7,175	1,691 ↑	FlightAware ADS-B (KLSV)
Fri 08:03:38 PM	35.9771	-115.2983	↓ 200°	268	8,575	3,433 ↑	FlightAware ADS-B (KLAS)
Fri 08:03:58 PM	35.9537	-115.3035	↓ 186°	276	9,750	2,820 ↑	FlightAware ADS-B (KLAS)
Fri 08:04:28 PM	35.9138	-115.3083	↓ 187°	308	10,925	1,898 ↑	FlightAware ADS-B (KLAS)

(c) Flight tracking information

Figure 5.10: Live flight tracking service by FlightAware

ATS route is typically created by linking the designated points (i.e., established waypoint-to-waypoint route). For example, Figure 5.11 shows an example of both designated points and ATS routes. The ATS routes and the designated points were downloaded from the FAA Aeronautical Data Delivery Service [97] to establish the U.S. airspace infrastructure used for this research.

GLOBAL_ID	LATITUDE	LONGITUDE	TYPE_CODE
E46C8B65-3E69-4606-AD51-010C56EEA93C	33-44-15.160N	111-07-40.620W	RPT
56D0911D-60F5-4D37-BBC1-C7AAA9CDAAF7	34-27-01.970N	109-02-52.770W	RPT
C08E5A7E-63A1-478F-A98C-9A867954DEB6	36-33-34.520N	112-11-59.850W	RPT
0CAADC33-B198-4F8C-9533-A07C25633EEA	34-35-00.720N	110-56-37.550W	RPT
E87C211E-4FC2-4D0A-A851-38AF12CD4EF8	33-06-20.100N	111-47-32.890W	RPT
19B6D0AC-23AE-408C-91B5-B1E1D0B517F9	32-03-56.360N	110-20-12.220W	RPT
470A5588-26B3-4029-A564-8E709DD7ED5D	32-00-00.000N	112-00-00.000W	NRS
6E6DEAEB-AB7A-43C8-A27D-046FB8B180DC	32-00-00.000N	110-00-00.000W	NRS
6D72555D-9230-4378-A753-EF539B22A38D	32-30-00.000N	112-00-00.000W	NRS
8CA7FD1E-DCA9-4670-947A-07F307D1F052	32-30-00.000N	110-00-00.000W	NRS
4AD5AC15-69E0-409F-91B1-974AFC471BAF	33-00-00.000N	114-00-00.000W	NRS

(a) Example of the designated points

TYPE_CODE	LEVEL	STARTPT_ID	ENDPT_ID
RNAV	L	8AEB9E1A-2090-464A-B076-8C37FE3F8B80	D7476D54-6DAA-4C55-9647-0A92930175A0
RNAV	L	D7476D54-6DAA-4C55-9647-0A92930175A0	B5BF6D95-D2B9-43B3-BE73-3F0F8144ECFF
RNAV	L	D52EB02F-4A1A-462A-AC57-A85C42A398EF	6E74703C-DD47-4CCA-996A-CCDA15636A3B
RNAV	L	F318D223-1F9C-4C5A-B108-0870A457D80C	B5BF6D95-D2B9-43B3-BE73-3F0F8144ECFF
RNAV	L	2BD4BAC2-77F5-49A9-AEFA-888790E551F7	71F5F428-928F-471B-9FAA-DE7632C00585
RNAV	L	71F5F428-928F-471B-9FAA-DE7632C00585	BDC4B4D5-E410-4894-9940-6ED2B83DD071
RNAV	L	B5BF6D95-D2B9-43B3-BE73-3F0F8144ECFF	D5EB6FC9-A607-45C2-9841-CD1D89D6B03B
RNAV	L	D5EB6FC9-A607-45C2-9841-CD1D89D6B03B	F5F0192D-83D5-4B71-A4C7-805CF1FDB32E
RNAV	L	F5F0192D-83D5-4B71-A4C7-805CF1FDB32E	609B9864-99FC-43EB-80B0-1411D2D90A20
RNAV	L	609B9864-99FC-43EB-80B0-1411D2D90A20	40874E08-D44B-429E-BBAA-B7ADCF45A0CE
RNAV	L	0698D77D-3A9C-4CBE-A8EC-6F4868A73841	F1E42B5C-5208-41C8-B410-320D915781CB

(b) Example of the ATS routes

Figure 5.11: FAA pre-determined designated points and ATS routes

5.2 Enabler 1: Supervised Machine Learning-based Wind Model

It is significantly important for a cockpit-based real-time flight path optimization framework to obtain wind information as accurately as possible because wind information is used to not only calculate how much fuel is required for a given flight but also optimize a flight route by seeking favorable winds.

This section presents a hybrid approach that combines a supervised machine learning algorithm with the Inverse Distance Weighting (IDW) technique to yield a continuous wind

prediction model in a more accurate manner. The hybrid approach consists of two parts that include 1) spatial regression and 2) temporal interpolation. Additional details will be discussed below.

5.2.1 Spatial Regression

First, the MLP was implemented to create a non-linear regression model of the GFS wind data with the aim of finding the best weight parameters to minimize errors between predicted and target values by adopting an iterative procedure. Figure 5.12 shows a flowchart of the MLP-based wind regression modeling process implemented in this research. To find the best weight parameters in the model, the Adam algorithm [98], which is an extended version of the stochastic gradient descent method, was implemented. The MLP-based wind regression model entails the following fully-connected layers as shown in Figure 5.13: 1) an input layer to receive the GFS wind data, 2) an output layer with the linear activation function that makes a prediction, and 3) two hidden layers with the sigmoid function.

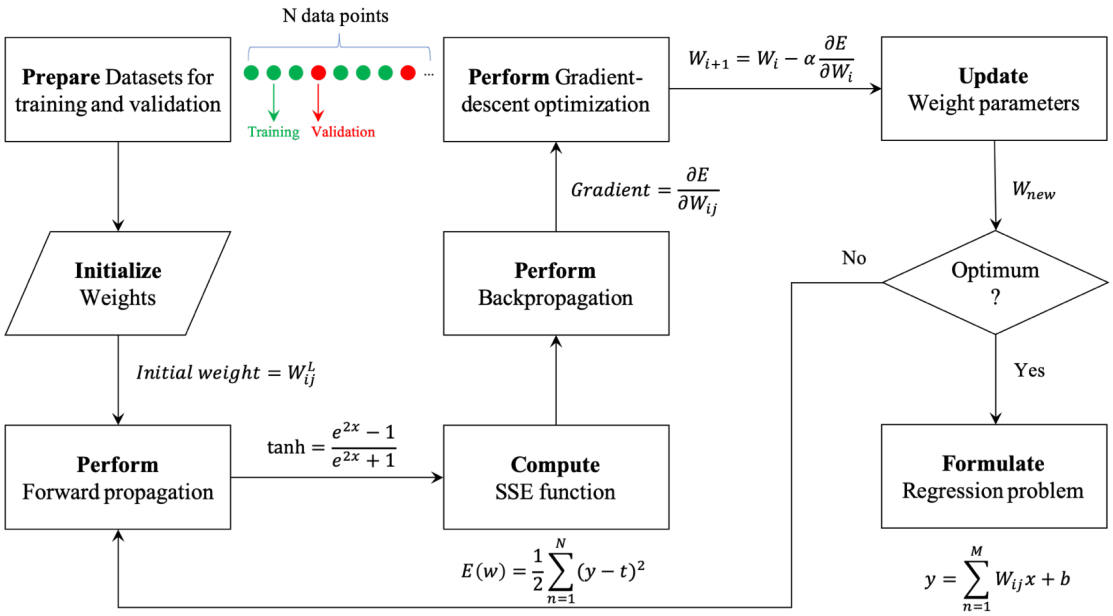


Figure 5.12: Flowchart of the MLP-based wind regression modeling process

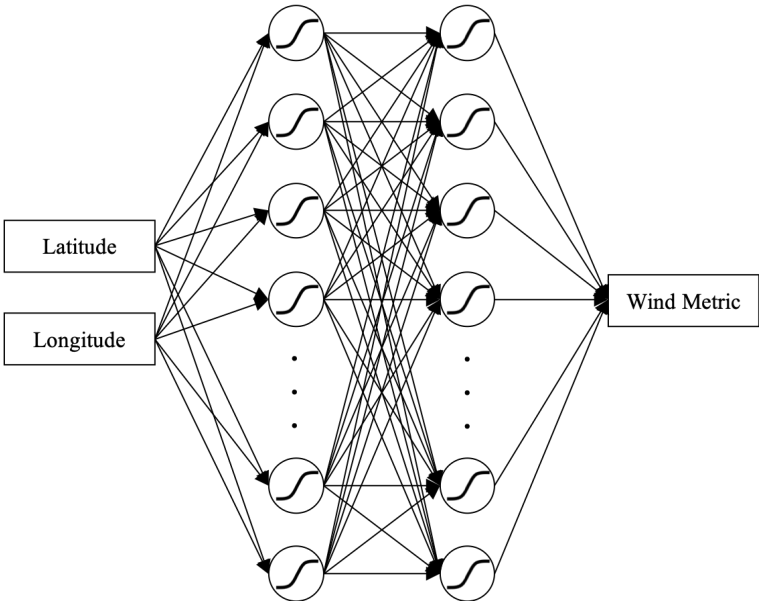


Figure 5.13: Diagram of the MLP-based wind regression model structure

To isolate the free hyperparameters the MLP wind regression model, the hybrid Design of Experiment (DoE) that consists of the two most well-known DoE methods was used. First, this research employed the Latin Hypercube Sampling (LHS) method to explore the inner points of the design space. As the LHS method typically does not capture corner points in a design space, this research also utilized the Central Composite Design (CCD) method with three factors to explore the corner points of the design space.

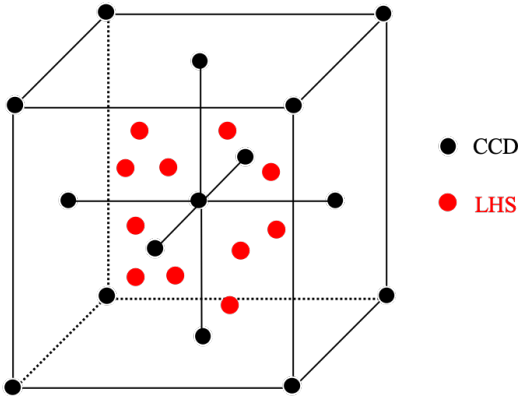


Figure 5.14: Notional sketch of the hybrid DoE

Figure 5.14 notionally shows how samples generated by the DoE methods are distributed in the design space with respect to a number of hidden layers, a number of hidden nodes, learning rate, regularization penalty parameter, and batch size. The effective MLP model was finally determined by the choice of hyperparameters tabulated in Table 5.2. Figure 5.15 shows the loss curve plot for the converged case with the chosen hyperparameters.

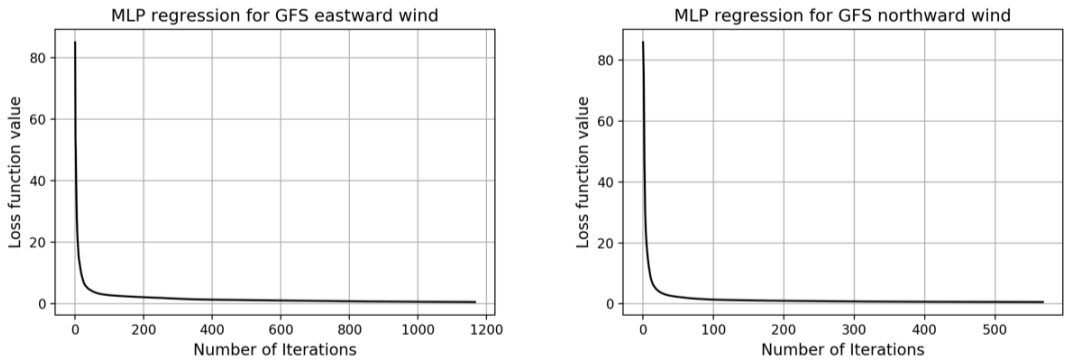


Figure 5.15: Loss curve for the final MLP wind regression model

Table 5.2: Design of Experiment results for the hyperparameters of the MLP model

Hyperparameter	Final choice
Number of hidden layers	2
Number of hidden nodes	50
Learning rate	0.0001
Regularization penalty parameter	0.001
Batch size	200

Second, the SVR [99] was implemented to obtain continuous values of the GFS eastward and northward wind in an entire two-dimensional space. Since the GFS wind data was not linearly separable in the original feature space, a few valid kernel functions (i.e., symmetric and positive semi-definite) were evaluated to transform the feature space. The RBF kernel function was finally chosen as it performed better than the other kernel functions.

The choice of the RBF kernel function naturally led to consider two hyperparameters of the algorithm that were the penalty parameter C and the RBF-related parameter r as they had greatly impacted the model performance. The penalty C parameter was controlled to balance between correct and incorrect data points in the objective function of the optimization problem. In addition, the r parameter was tuned to ensure that the algorithm was not too constrained and affected by an over-fitting issue. In particular, a grid search method as notionally shown in Figure 5.16 was implemented to find out the best values for the hyperparameters. The effective SVM algorithm, especially for given GFS wind data, was finally determined by the choice of hyperparameters.

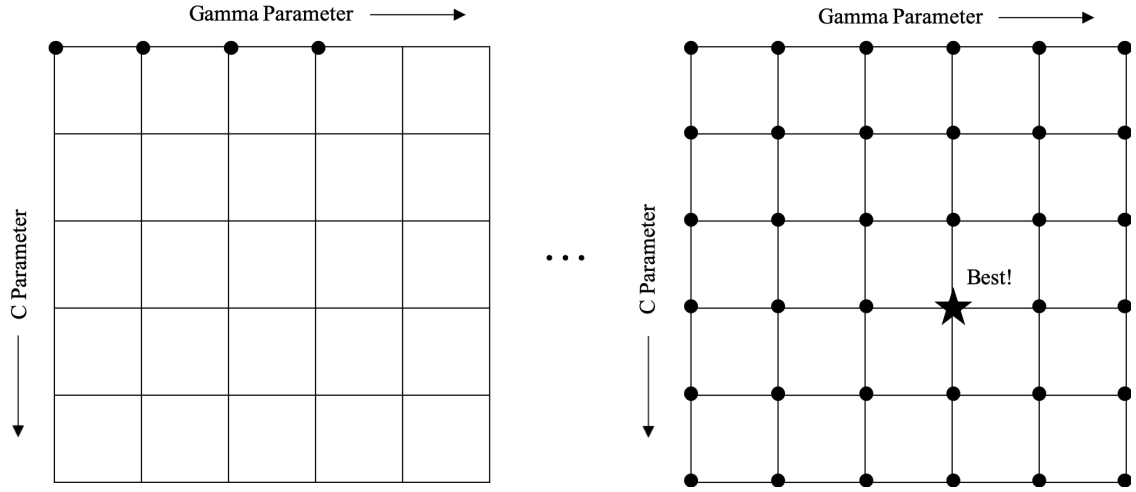


Figure 5.16: Notional sketch of a grid search approach for SVR hyperparameter selection

Third, the GP was implemented to create a non-linear probabilistic regression model of the GFS wind data. Given N input and output pairs, the squared-exponential kernel function, which is defined as $k(x_i, x_j) = \theta_0 \exp(-\frac{\theta_1}{2}\|x_i - x_j\|^2)$, was utilized during the GP regression process where the hyperparameters of the kernel were optimized by maximizing the log-marginal-likelihood of the outputs.

5.2.2 Temporal Interpolation

As the GFS weather data is available with hourly intervals (e.g., 12:00 and 13:00 UTC), given that wind information at a particular timestamp (e.g., 12:15 UTC) is required for the flight path optimization module (i.e., *PATH.py*) developed in this research, a temporal interpolation model on the GFS wind data was created with the IDW technique in order to estimate the wind speed and direction at any specific time within an hour. The IDW is a mathematical model, which is based on the fact that closer values are more related than further values, defined in Equation 5.1 and Equation 5.2.

$$wind(t) = \frac{\sum_{i=1}^{N} w_i(t) wind_i}{\sum_{i=1}^{N} w_i(t)} \tag{5.1}$$

$$w_i(t) = \frac{1}{d(t, t_i)^p}, \tag{5.2}$$

where N is total number of known points, p refers to power parameter, d represents distance between given points, t_i is a known point, and t is an interpolated point. The sigma notation basically means that the number of known points can be controlled if needed for interpolating the point. For example, if it defines $N = 2$, then it uses the two closest known points for the interpolated point. It is important to note that $wind(t)$ becomes $wind_i$ if $d(t, t_i) = 0$ for some i, meaning that the interpolated point is exactly matched to the known point. The nearest known point value is given for the interpolated point value. On the other hand, $wind(t)$ is estimated by Equation 5.1 and Equation 5.2 if $d(t, t_i)$ is not equal to zero for all i. This estimation does have a value that is neither above the maximum nor below the minimum known values.

This research implemented the IDW method particularly with the assumption, claiming that closer wind values have more influenced on the interpolated point than wind values that are farther away (e.g., wind values at 12:10 UTC are more close to wind values at 12:00 UTC than wind values at 13:00 UTC). The Epoch time, which is the number of

seconds that have elapsed since January 1st, 1970, midnight UTC, was used to compute the distance in Equation 5.2. Given a set of known wind values, a wind value at a certain point with respect to timestamp was estimated. A high-level summary of the implementation is presented in Algorithm 2. Figure 5.17 depicts how to perform the IDW with the basic form defined by Shepard [100].

Algorithm 2 IDW Function in *WIND.py*

```
Define power parameter
Convert time information into the Epoch time
Specify current and next time
Specify wind space regression values
Compute distance between current and next Epoch time information
function IDW(Power Parameter, Distance, Wind Space Regression Values)
    if distance=0 then
        Define a wind value using the nearest linear function
    else if distance!=0 then
        Calculate the weight (w_i)
        Perform GFS time interpolation using the IDW method
    end if
end function
```

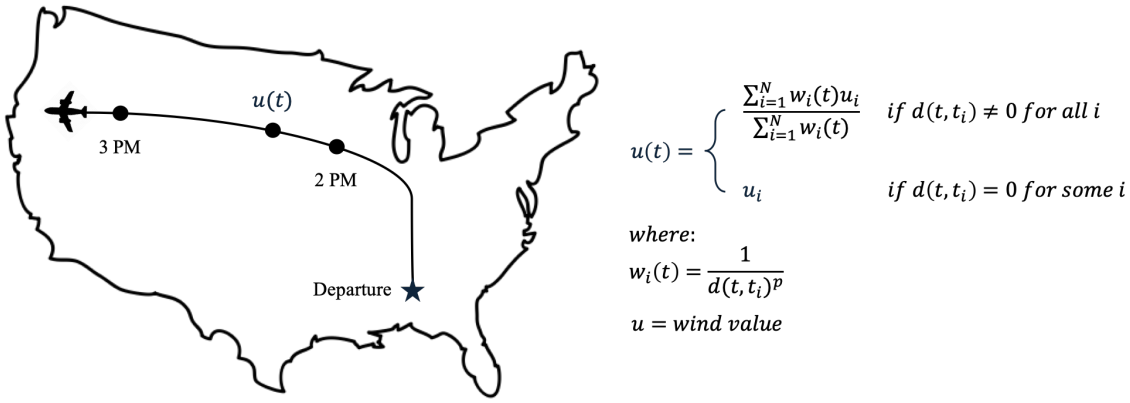


Figure 5.17: IDW method for GFS wind temporal interpolation

5.2.3 Research Experiment

To satisfy the first requirement (i.e., the capability of providing weather information accurately and continuously to a cockpit-based real-time flight path optimization framework within a specified time) raised by the Research Question 1, this research specifically constructed the Research Hypothesis 1.1 as follows:

> *Research Hypothesis 1.1: A supervised machine learning-based regression method will provide wind forecasts with a lower prediction error (i.e., RMSE) compared to a linear interpolation method.*

This section introduces the Research Experiment 1.1 to investigate and test the Research Hypothesis 1.1. The purpose of this research experiment is to evaluate model performance for both supervised machine learning-based wind regression models and a linear interpolation model in order to eventually identify the most appropriate wind prediction model used for a cockpit-based real-time flight path optimization framework.

The procedure implemented for this research experiment is described as follows: Step 1 is to evaluate three different supervised machine learning-based wind regression models (i.e., MLP, SVM, and GP) to identify the most appropriate wind prediction model. It is important to note that Step 1 does not compare any supervised machine learning-based wind regression models with a linear interpolation method. Step 2 is to compare supervised machine learning-based wind regression models with a linear interpolation model in order to substantiate the Research Hypothesis 1.1. Given that the Research Hypothesis 1.1. is demonstrated to be true, Step 3 is to compare wind values predicted by the selected machine learning-based wind regression model with actual wind measurement datasets observed by weather balloons. Additional details will be discussed below.

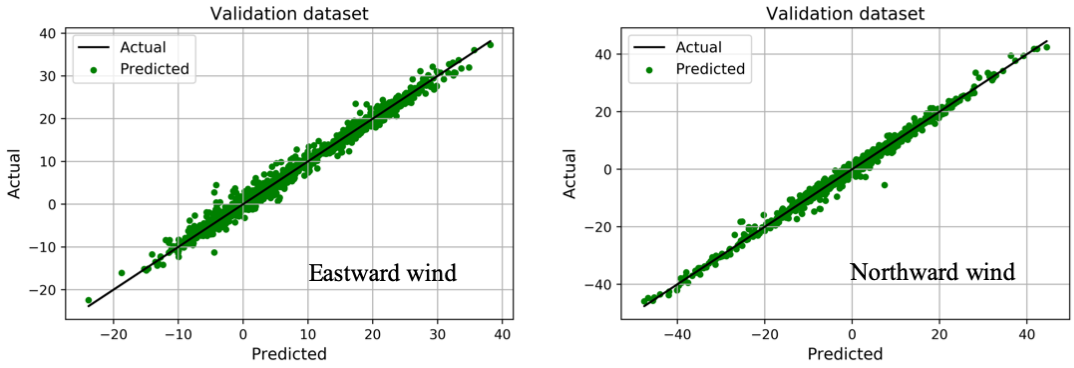
Step 1. supervised machine learning-based wind regression model evaluation

To identify the most appropriate algorithm for spatial regression, the coefficient of determination (i.e., R-squared) for three different models was computed. As can be seen in Table 5.3, the R-squared results show that the SVM provided better prediction compared to the other models. However, it is important to note that a high value of the R-squared does not imply that the model is accurate; but it only provides an indication as to whether or not the next step of evaluating model performance can be progressed.

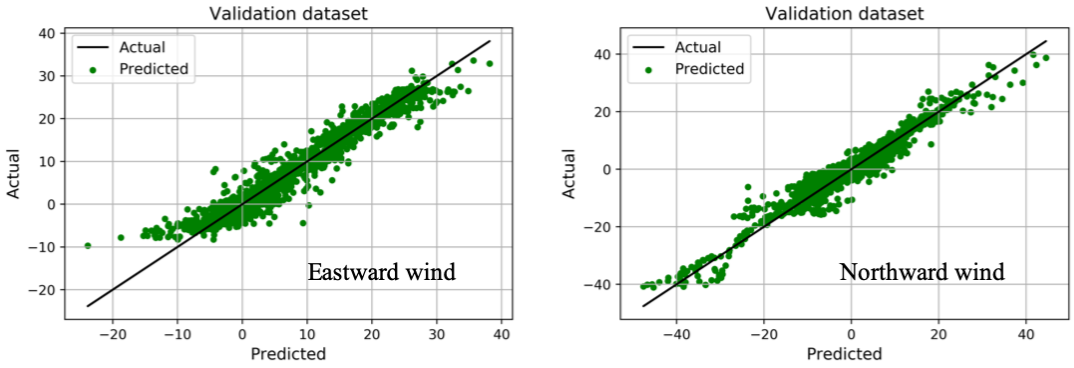
Table 5.3: Supervised machine learning-based wind regression model evaluation

	RMSE (m/s)		R-squared	
Model	Eastward Wind	Northward Wind	Eastward Wind	Northward Wind
SVM	1.39	1.28	0.99	0.99
MLP	3.65	3.94	0.95	0.94
GP	1.88	1.83	0.97	0.98

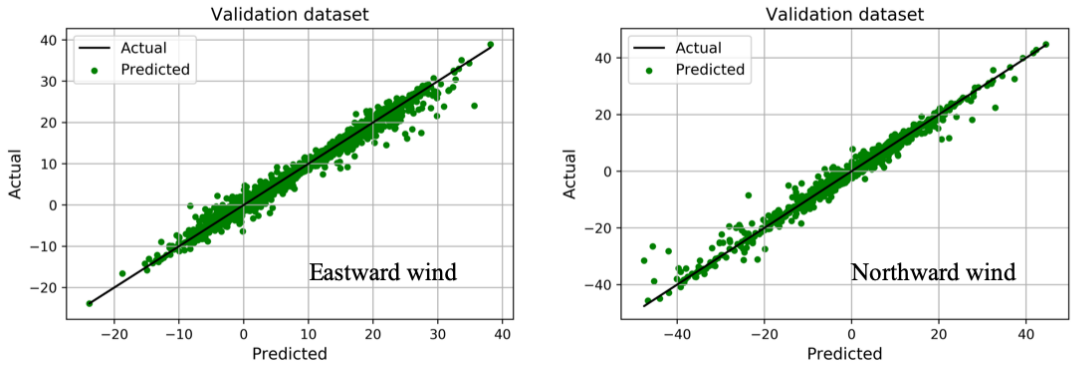
As the next step of model evaluation, the RMSE was calculated to estimate the standard deviation of the random error (i.e., how residuals are spread out). In particular, two different types of RMSE distributions, which involve 1) Model Fit Error (MFE) that represents how well the machine learning model fits the wind data points and 2) Model Representation Error (MRE) that represents how well the machine learning model predicts the actual response, were investigated. Since the MFE (i.e., training error) was not sufficient for the model evaluation process, this research specifically focused on the MRE (i.e., validation error) to ensure the predictive capability of the supervised machine learning-based wind regression models. The MRE results are summarized in Table 5.3, indicating that the SVM model performed better than the other models. Figure 5.18 shows the results of the MREs for the supervised machine learning-based wind regression models.



(a) Support Vector Machine (SVM)



(b) Multi-Layer Perceptron (MLP)



(c) Gaussian Process (GP)

Figure 5.18: Actual vs. Predicted plots of the supervised machine learning-based wind regression models

Step 2. supervised machine learning-based regression vs. linear interpolation

Validation points were randomly sampled as shown in Figure 5.19 to evaluate the performance of two approaches, namely supervised machine learning-based regression (i.e., Enabler 1) and linear interpolation, by comparing with the GFS wind values at the validation points. Three different values at the validation points, which are 1) GFS wind value, 2) wind value estimated by Enabler 1, and 3) wind value estimated by the linear interpolation method, were obtained for the comparison.

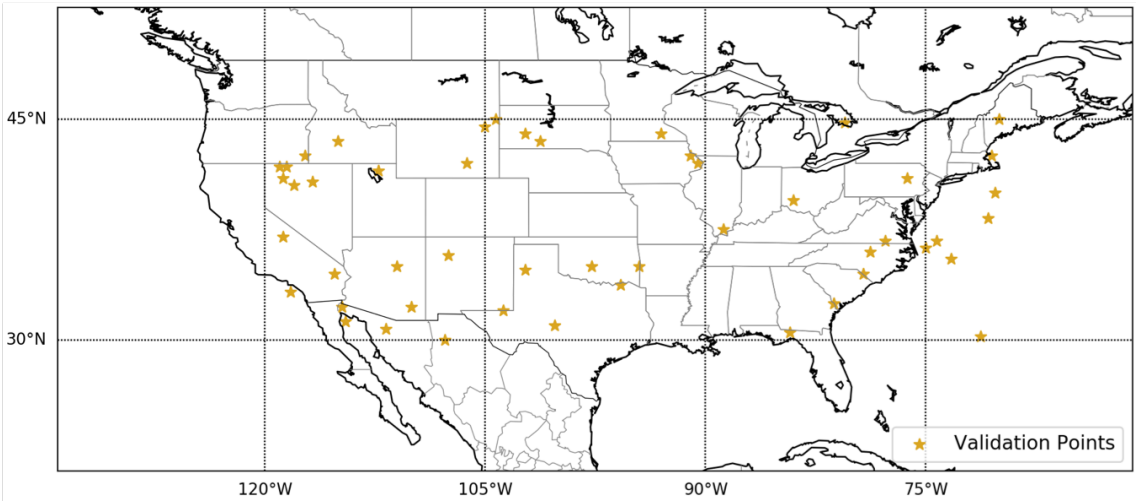


Figure 5.19: Validation points for the comparison between linear interpolation and supervised machine learning-based regression

Table 5.4: Supervised machine learning-based regression vs. Linear interpolation

Method	Eastward wind RMSE (m/s)	Northward wind RMSE (m/s)
Support Vector Machine	0.79	0.87
Multi-Layer Perceptron	1.14	1.36
Gaussian Process	0.91	1.28
Linear Interpolation	1.21	1.45

Table 5.4 shows the results of the RMSE for three different machine learning models and the linear interpolation model, indicating that all of the supervised machine learning-based regression methods provided better wind prediction compared to the linear interpolation method for the validation points.

Step 3. supervised machine learning-based regression vs. weather balloon

Actual wind measurement data, which is typically observed by weather balloons (i.e., radiosonde) [101], was collected to evaluate the accuracy of the selected supervised machine learning-based wind regression model (i.e., SVM). Weather balloons are launched at about 100 stations in the U.S. with attaching a parachute. Once weather balloons are launched, they travel thousands of feet of atmosphere to gather critical weather information. It basically provides a primary source of upper-air observation data that includes pressure, temperature, wind direction and speed, and relative humidity. These observations are controlled/monitored by weather forecasters and reported twice a day (i.e., 00:00 and 12:00 UTC) for general public.

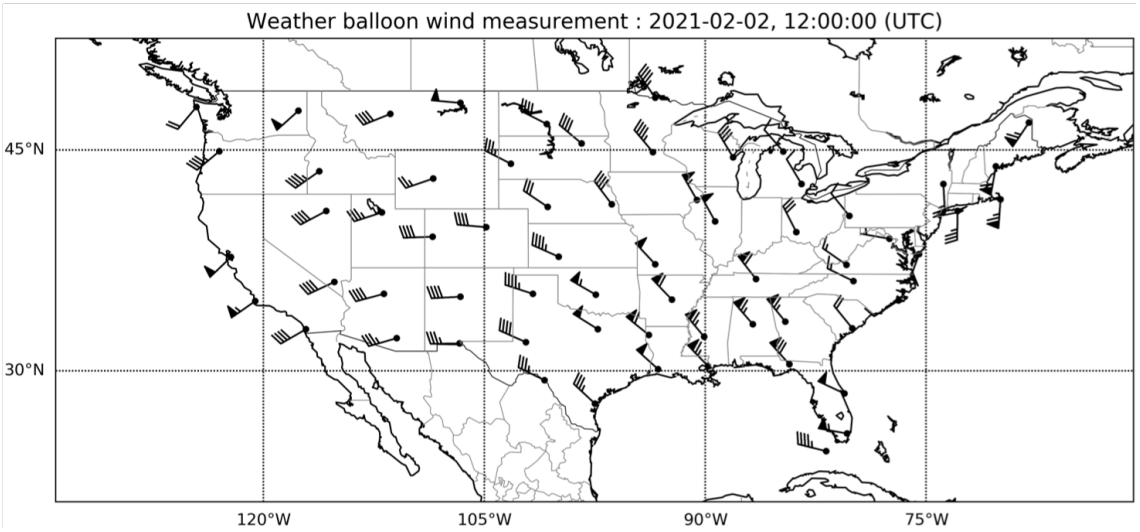


Figure 5.20: Wind measurement data by the weather balloons at 2021-02-02 12:00 UTC (Altitude = 250 hPa)

Figure 5.20 shows two-dimensional graphics of wind barb information across the U.S., especially measured by the weather balloons on February 2nd, 2021. It is important to note that this research only connected the stations where wind information was available at the specific date. More specific information regarding wind measurement data is found on Appendix B.

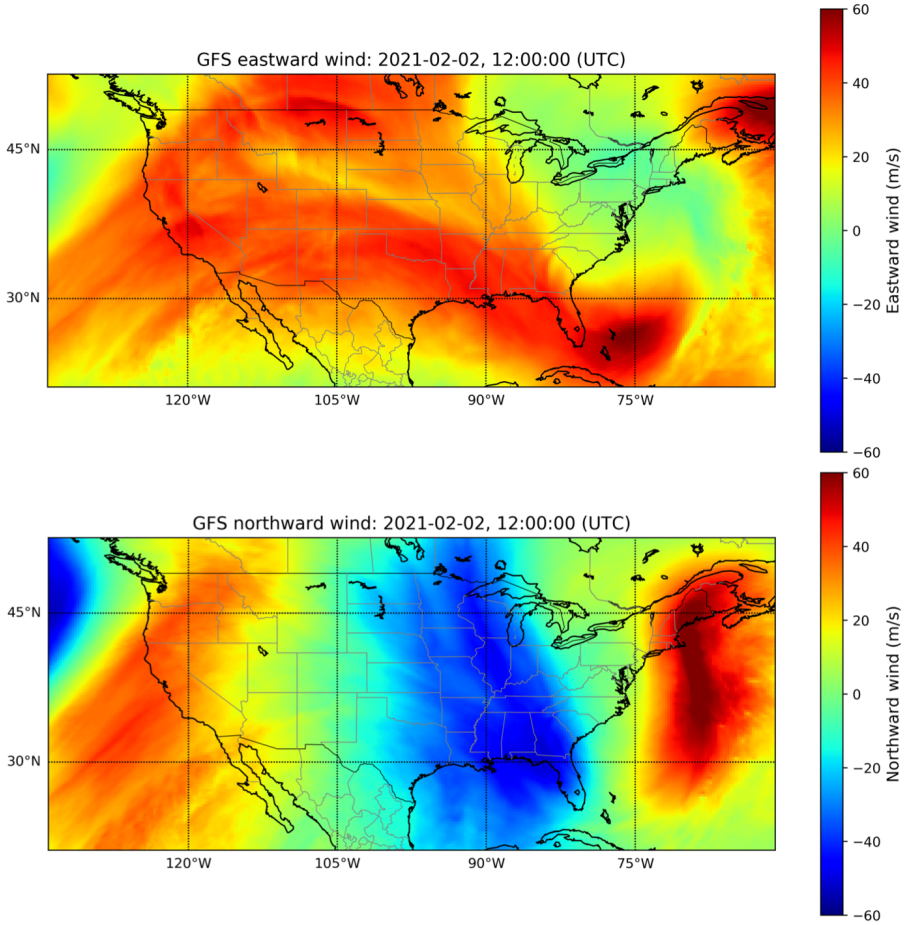
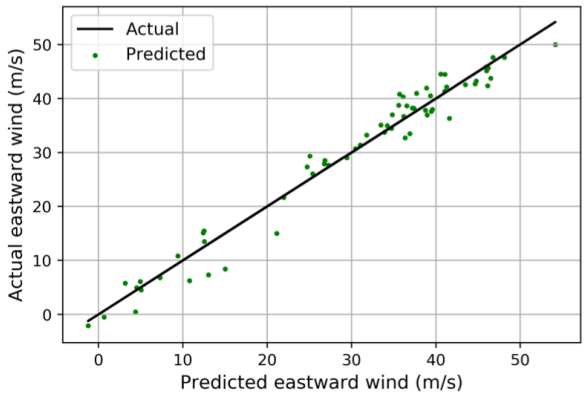


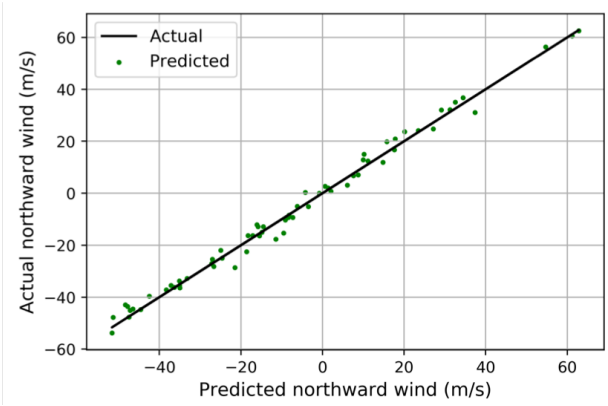
Figure 5.21: GFS wind visualization at 2021-02-02 12:00 UTC (Altitude = 250 hPa)

To compare actual wind measurement data with wind values predicted by Enabler 1, the GFS wind data at the specific time (i.e., February 2nd 12:00 UTC) and altitude (i.e., 250 hPa) was particularly downloaded to be consistent with the observational data obtained by the weather balloons. Figure 5.21 shows a visualization of eastward and northward

GFS wind data at the specific time and altitude. The following data pre-processing steps were conducted after downloading the GFS wind datasets: 1) decoded the original GFS wind data, 2) reconstructed the wind datasets for a supervised machine learning process, 3) decomposed the wind datasets into training and validation datasets, and 4) trained the wind datasets to generate a regression model. The regression model was then used to predict wind values at the stations where the weather balloons were launched. Figure 5.22 shows the results of actual by predicted plots for both eastward and northward wind values.



(a) Actual by Predicted (Eastward wind)



(b) Actual by Predicted (Northward wind)

Figure 5.22: Actual wind measurement vs. Machine learning-based wind prediction

The RMSE for both eastward and northward cases was calculated. As a result, the RMSE for eastward wind was 2.7 m/s and the RMSE for northward wind was 2.2 m/s. The errors seem relatively either large or small depending on circumstances; however, it

appears that Enabler 1 (i.e., supervised machine learning-based wind model) reasonably provided wind predictions over the entire U.S.

5.3 Enabler 2: Unsupervised Machine Learning-based Convective Weather Model

It is imperative for a cockpit-based real-time flight path optimization framework to not only visualize graphical areas of the convective weather forecasts but also implement convective weather forecast activity fields (e.g., grid-based polygon areas) in the framework because pilots encounter convective weather every day when they are in flight.

This section presents an unsupervised machine learning-based short-term (i.e., every 10 minutes) convective weather modeling approach using multiple observational datasets (e.g., NEXRAD, METAR, and PIREP) to forecast reliable and up-to-date convective weather activity. An overview of the unsupervised machine learning-based short-term convective weather modeling approach [35] is described in Figure 5.23. Additional details will be discussed below.

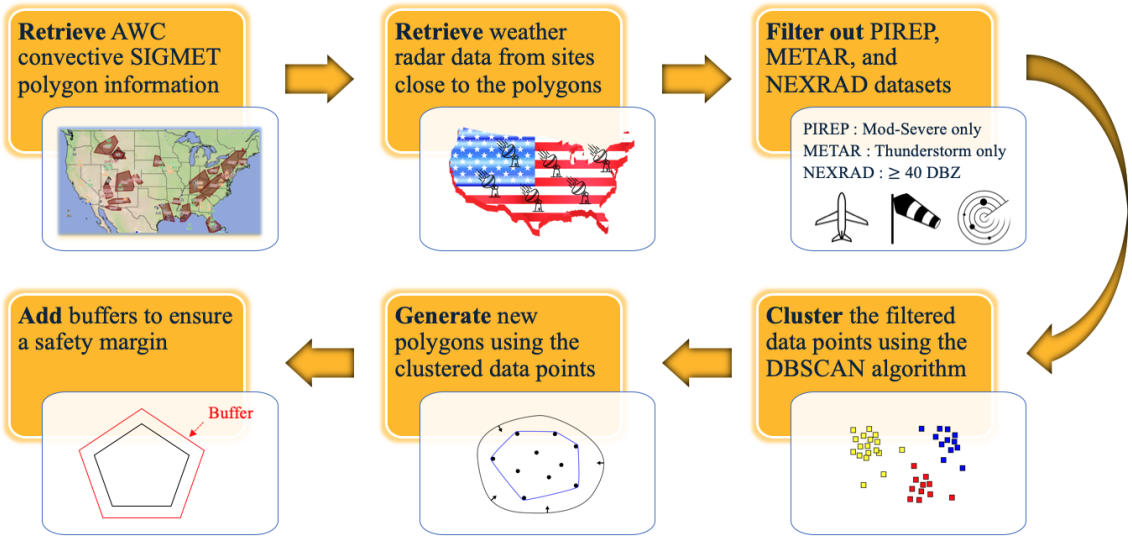


Figure 5.23: Overview of the unsupervised machine learning-based convective weather polygon generation process

5.3.1 Data Collection

The process starts by connecting the TDS to retrieve the latest convective SIGMET polygons issued by the AWC. Once the algorithm identifies current convective SIGMET polygons, it connects the radar sites close to the polygons and downloads multiple observational datasets that include NEXRAD, METAR, and PIREP. The fact that the algorithm collects NEXRAD datasets from radar sites only close to the latest convective SIGMET polygons is particularly significant given that the algorithm needs to reduce computational costs as much as possible. Figure 5.24 shows selected NEXRAD site locations at a specific time.

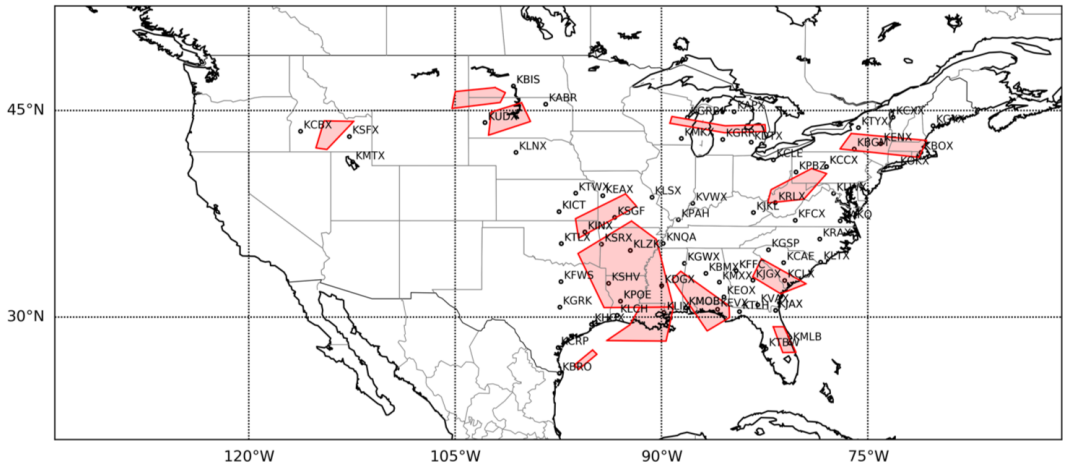


Figure 5.24: Selected NEXRAD site location visualization

5.3.2 Data Filtering

The algorithm is designed to perform additional data pre-processing steps after automatically collecting datasets. For example, the algorithm filters out 1) NEXRAD data above 40 dBZ representing the level of moderate-to-heavy intensity [72], 2) METAR data with only thunderstorm reports, and 3) PIREP data with only moderate or severe turbulence. Once the filtering process is completed, the algorithm computes a center for the polygons and evaluates distances between the centers and all of the filtered points (i.e., current points).

The points are then re-configured based on the moving speed and direction information of the SIGMET polygons. Figure 5.25 shows an example visualization of filtered METAR reports at a specific time. All of these filtered data points are handed over to the next step, which is data clustering.

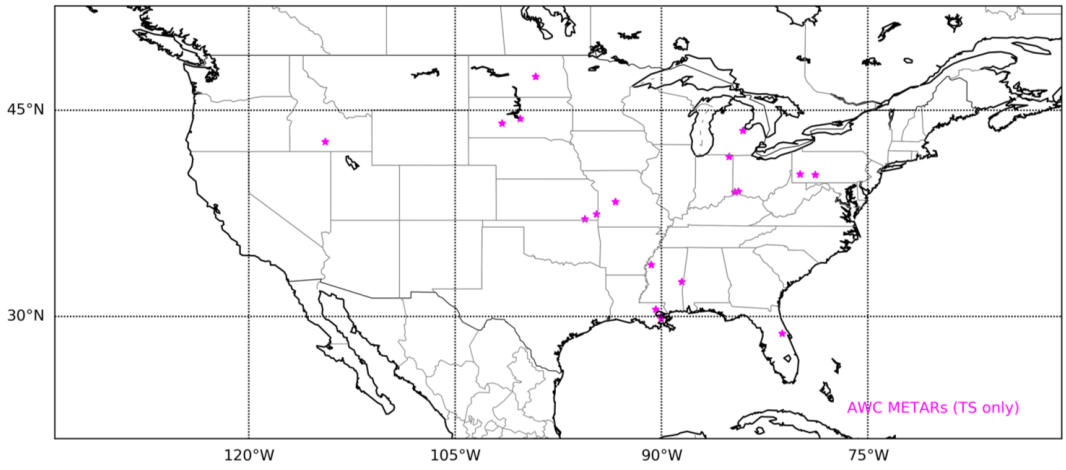


Figure 5.25: Filtered METAR report visualization

5.3.3 Data Clustering

The algorithm employs an unsupervised machine learning algorithm, namely the DBSCAN, to cluster the filtered data points. These clustered data points are used to generate polygons delineating convective weather activity. There are several reasons why the DBSCAN algorithm is selected for this research instead of other clustering algorithms such as either K-Means or EM algorithm. First, the algorithm does not require to specify the number of clusters. This is significant because the weather domain typically requires minimum knowledge. Second, the algorithm is robust to outliers. This is also important because the weather domain normally contains datasets that are noisy and irregular. Figure 5.26 shows an example of the DBSCAN results for the filtered data points.

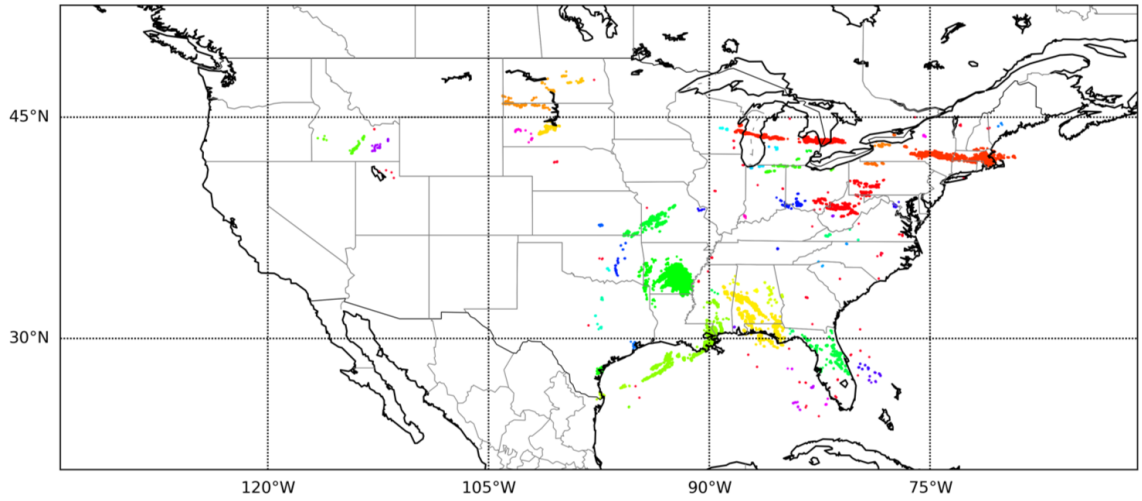


Figure 5.26: DBSCAN results for the filtered data points

5.3.4 Polygon Generation

Given that there must be a way that accounts for transferring multiple observational datasets into some measurable shapes used for the flight path optimization module (i.e., *PATH.py*) developed in this research, the algorithm generates a convex hull for the clustered data points, adds lateral buffer layers to ensure a safety margin, and remove polygons if the areas of the polygons are smaller than the user-defined area threshold (i.e., $500km^2$).

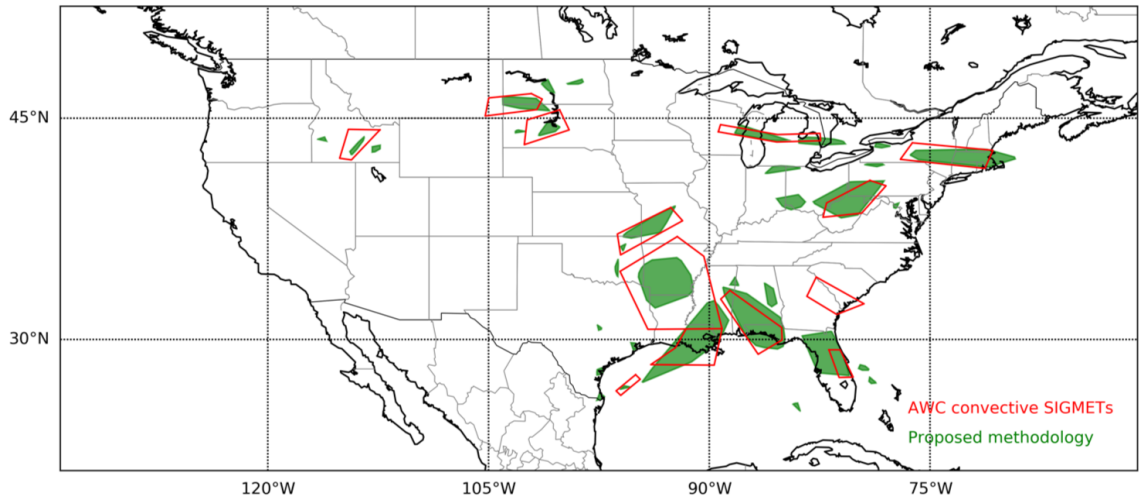


Figure 5.27: Polygon generation with the clustered data points

In particular, the buffer layer is determined based on interview surveys, operational manuals, and feedback from the airline pilots. For example, according to the Boeing 777 Quick Reference Handbook [102], severe weather detected by the on-board radar should be avoided by at least 10 nautical miles at or below an altitude of 20,000 feet and by a minimum of 20 nautical miles at an altitude above 20,000 feet. This type of information is specified as a user-defined variable to determine the size of the buffer layer in the framework developed for this research. Figure 5.27 shows the comparison between new polygons (i.e., green polygons) generated by the unsupervised machine learning-based short-term convective weather model and the AWC convective SIGMET polygons (i.e., red polygons).

5.3.5 Research Experiment

To satisfy the first requirement (i.e., the capability of providing weather information accurately and continuously to a cockpit-based real-time flight path optimization framework within a specified time) raised by the Research Question 1, this research specifically constructed the Research Hypothesis 1.2 as follows:

> *Research Hypothesis 1.2: An unsupervised machine learning-based clustering algorithm with multiple observational datasets which are updated every 10 minutes will define the areas of convective weather activity more accurately than the AWC convective weather data by generating convective weather polygons in a more frequent manner.*

This section introduces the Research Experiment 1.2 to investigate and test the Research Hypothesis 1.2. The purpose of this research experiment is to visually compare new polygons generated by the unsupervised machine learning-based short-term convective weather model with the AWC convective SIGMET data. The procedure implemented for this research experiment is described as follows: Step 1 is to collect all of the multiple observational datasets such as NEXRAD and METAR. Step 2 is to visualize polygon areas

issued by the AWC. Step 3 is to visualize polygon areas that are generated by the unsupervised machine learning-based short-term convective weather model. Step 4 is to visually compare the AWC convective SIGMET polygons with the new polygons.

As a part of this research experiment, actual NEXRAD information on February 12th, 2021 was retrieved to compare the AWC convective polygons with new polygons generated by the unsupervised machine learning-based short-term convective weather model. Figure 5.28 shows the comparison between the AWC convective SIGMET data and the new polygons generated by the unsupervised machine learning-based short-term convective weather model based on the corresponding NEXRAD data information.

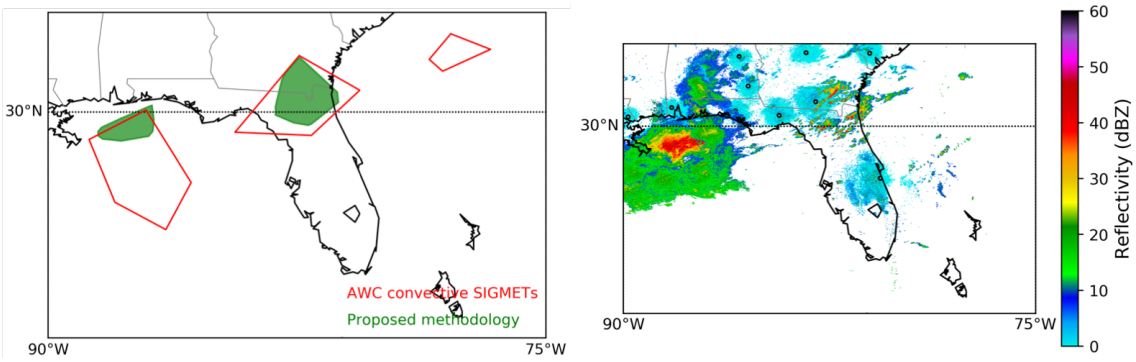


Figure 5.28: NEXRAD and polygon visualization at 2021-02-12 20:00 UTC

Given that NEXRAD information is a reliable ground-based observational dataset, it seems that the green polygons are more matched with the NEXRAD dataset compared to the red polygons. This means that the red polygons might be drawn by the weather forecasters in a more conservative manner to reduce risks as much as possible. A conservative approach is important especially in the aviation industry; however, this does not mean that pilots are not able to make a tactical decision along the flight trajectory. This is significant from the perspective of simulation as the framework developed in this research finds an optimal flight trajectory based on polygon information. For example, if the red polygon shapes are given in the simulation environment, the framework has to avoid the polygon

areas completely as they are considered as hard constraints, leading to excessive deviations compared to reality. On the other hand, once accurate polygon shape information is given in the simulation environment, it helps the framework avoid excessive deviation as well as find more acceptable flight routes.

To further investigate the fact that the AWC convective SIGMET data did not always accurately reflect the actual convective activities in the vicinity of hazard weather, one case study was performed with the previous American Airlines Flight 1300 as a proof-of-concept. The intent of this case study was to compare the AWC convective weather data with new polygons generated by Enabler 2 (i.e., unsupervised machine learning-based short-term convective weather model) along with the actual flight (i.e., AA1300) trajectory. Details of the steps are as follows: 1) collect the previous AA1300 Flight information (i.e., altitude, ground speed, latitude, longitude, and time) from FlightAware and the corresponding observational weather data that include PIREP, METAR, SIGMET, and NEXRAD, 2) generate new polygons with the observational datasets, and 3) compare visually the new polygons with the AWC convective SIGMET polygons along with the AA1300 flight trajectory. Figure 5.29 describes the scenario sequence of this case study.

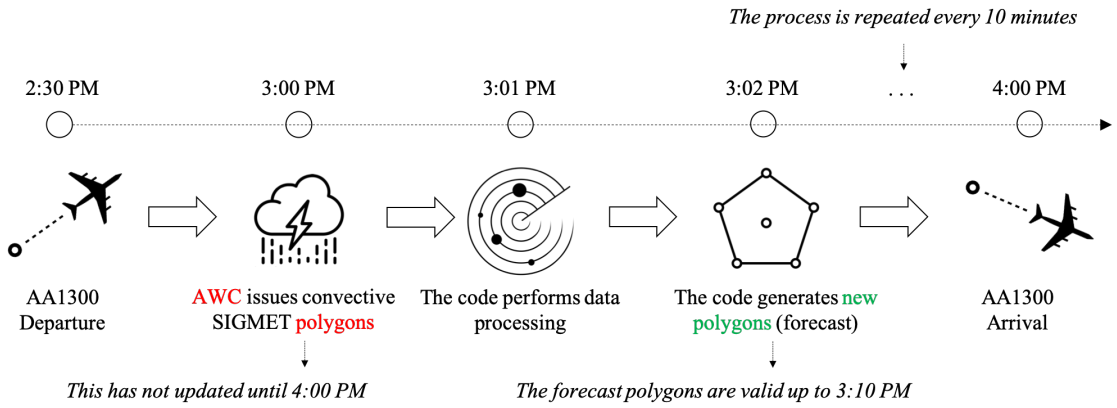


Figure 5.29: Scenario sequence of AA1300 case study

Figure 5.30 shows the result of the comparison between the AWC convective SIGMET

polygons and the new polygons generated by Enabler 2. The green line represents the trajectory of the previous AA1300 Flight used for this case study. As can be seen, Enabler 2 provided a better picture of the nearby convective weather activity compared to the AWC convective SIGMET polygons for this flight case. This result indicates that the AWC convective weather polygon areas could be broken into some measurable shapes without losing the key information of the AWC convective SIGMET data.

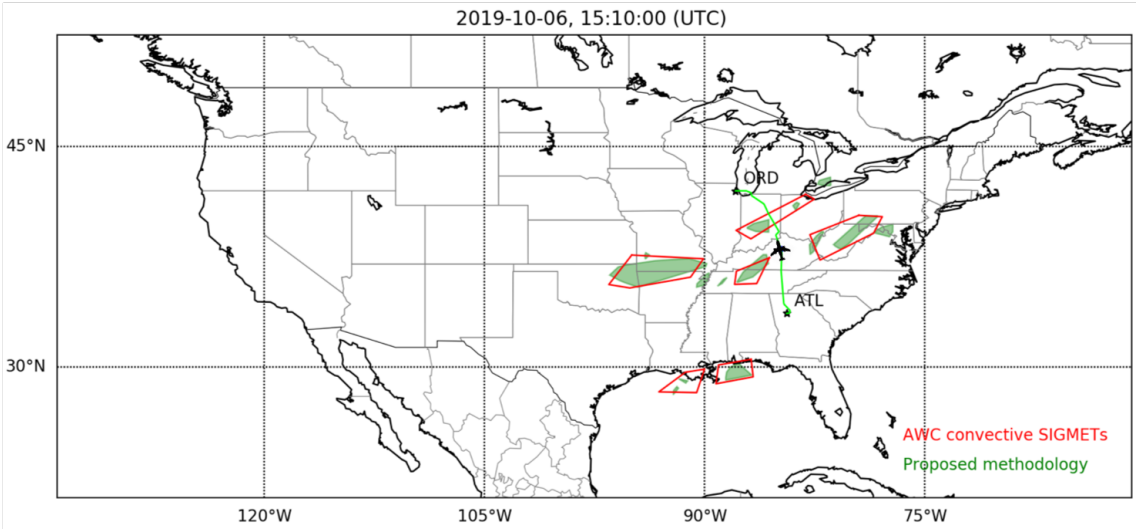


Figure 5.30: AWC convective SIGMETs vs. New polygons generated by Enabler 2

In a nutshell, Enabler 2 would allow the algorithm to avoid excessive deviation as it would not have to avoid the AWC convective SIGMET polygon areas completely but rather consider the new polygons in the computer simulation environment, which potentially helps the algorithm take a shorter flight route by tactically selecting more acceptable flight route opportunities such as flying between polygon areas.

5.4 Enabler 3: Designated Points-based Flight Path Optimization Model

While it is important for a cockpit-based real-time flight path optimization framework to model that aircraft follows established waypoint-to-waypoint flight routes to comply with

the ATC requirements, it is worth mentioning that pilots sometimes do not follow established waypoint-to-waypoint flight routes in reality but rather choose other flight route options as needed.

This section presents a designated points-based flight path optimization model that combines the A* search algorithm with a free-flight approach to provide flight route selections that are not necessarily constrained to established waypoint-to-waypoint flight routes but provide acceptable free-flight route options. Additional details will be discussed below.

5.4.1 Assumptions

The designated points-based flight path optimization model proposed in this research relies on the following assumptions: 1) an en-route phase is only considered (i.e., both departure and arrival phases are neglected), 2) aircraft true airspeed is constant during the en-route phase, 3) aircraft follows established waypoint-to-waypoint flight routes except in the situation where hazardous weather is encountered, 4) convective weather activity is represented by polygons associated with high penalty costs if aircraft penetrates them (i.e., polygons must be avoided in a computer simulation environment to reduce total travel time), and 5) TFRs are not considered.

5.4.2 Airspace Network Generation

To enable graph-based approaches to find optimal flight routes, it is imperative to define a network (i.e., nodes and edges) that models the architecture of U.S. airspace. The ATS routes and the designated points were downloaded from the FAA Aeronautical Data Delivery Service [97] to establish the U.S. airspace infrastructure.

The data pre-processing steps were specifically conducted with the following procedures: 1) filter out the ATS routes by only considering upper levels, 2) record start and end IDs for the filtered routes, 3) cross-reference the IDs with the designated points, and 5) specify start and end point coordinate information (e.g., longitude and latitude) for the

filtered ATS routes. Figure 5.31 shows the filtered ATS routes that connect the filtered designated points.

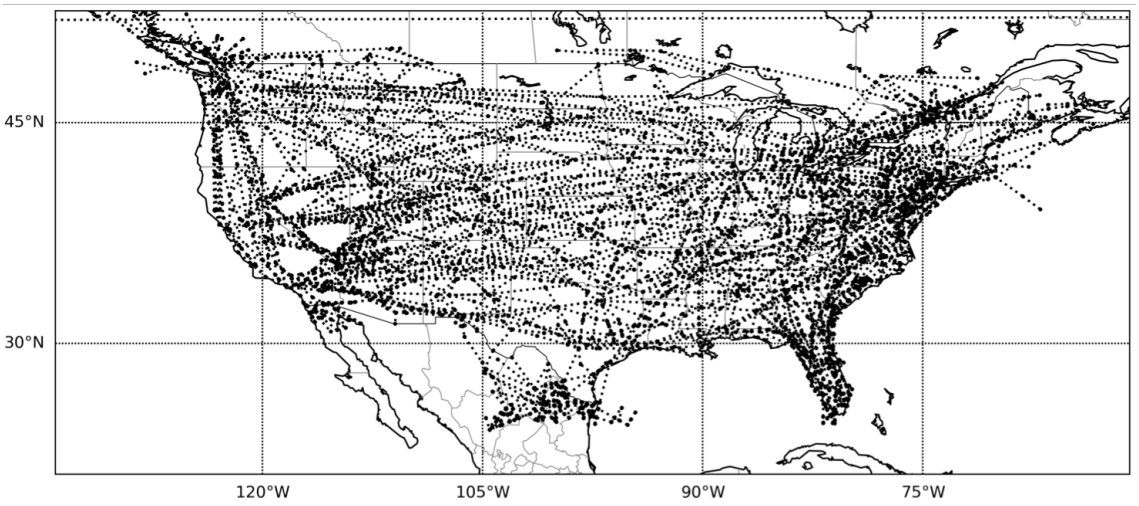


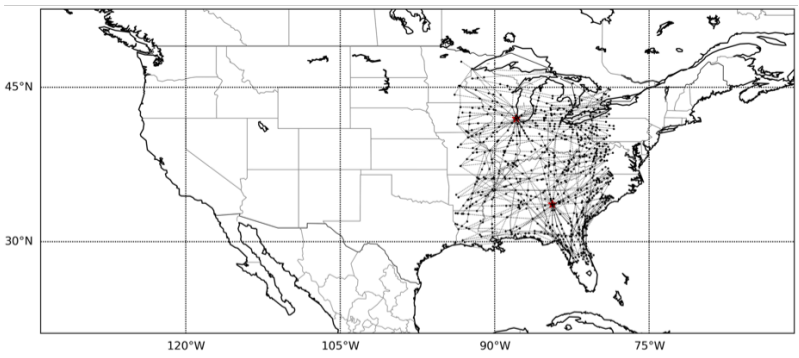
Figure 5.31: U.S. airspace infrastructure (high altitude only)

5.4.3 Bounding Box

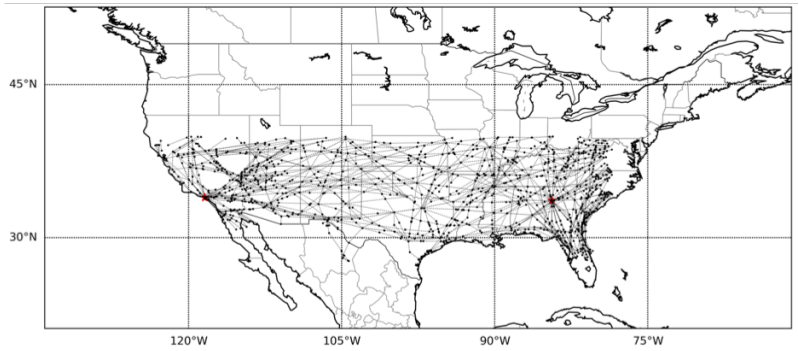
To account for the worst case where the designated points-based flight path optimization model suffers from computational costs, several features which were designed to not only eliminate the computational burden but also generate acceptable optimal solutions in a finite time were implemented. Creating a bounding box was one of the features that potentially reduced computational costs in some cases. A high-level summary of the implementation is presented in Algorithm 3.

Algorithm 3 Bounding Box Function in *PATH.py*

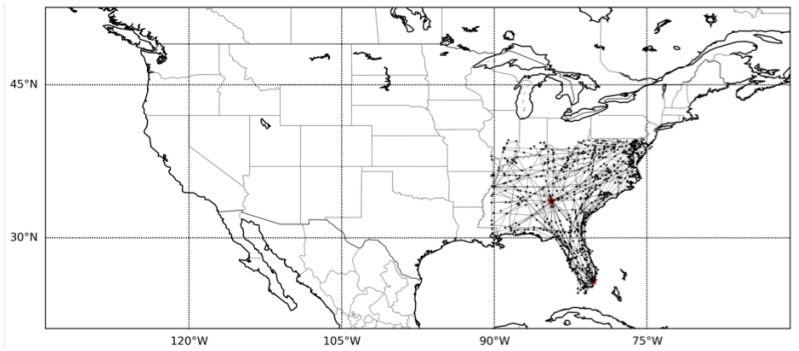
```
Specify the user-defined Margin with the format [Lon, Lat]
function BOUNDING BOX(Origin, Destination, Margin)
    Min. lon of bounding box = min(Origin lon, Destination lon) - Margin[0]
    Max. lon of bounding box = max(Origin lon, Destination lon) + Margin[0]
    Min. lat of bounding box = min(Origin lat, Destination lat) - Margin[1]
    Max. lat of bounding box = max(Origin lat, Destination lat) - Margin[1]
end function
```



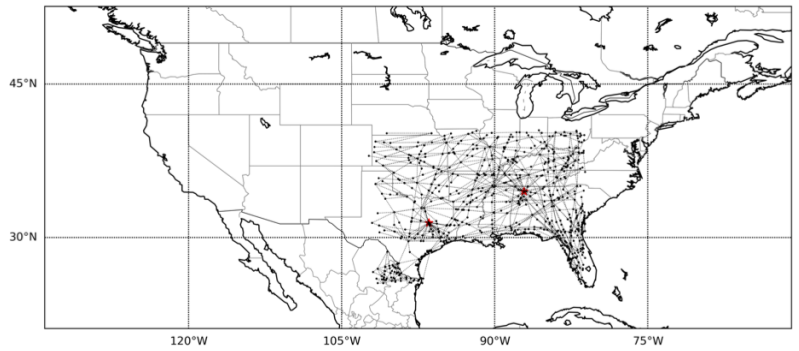
(a) ATL–ORD Flight



(b) ATL–LAX Flight



(c) ATL–MIA Flight



(d) ATL–IAH Flight

Figure 5.32: U.S. airspace infrastructure within the bounding box

It is important to note that the designated points-based flight path optimization model is designed to conduct the search only within the bounding box. For example, Figure 5.32 shows the filtered network where the algorithm is supposed to search on the network in order to find an optimal flight route. The performance of the designated points-based flight path optimization model is greatly impacted by the choice of an origin and destination. This implies that creating a bounding box may not be useful if the model analyzes a long cross-state flight such as from New York to Los Angeles.

5.4.4 Travel Time Estimation

To estimate travel time between established waypoint-to-waypoint flight routes, a Python code (i.e., *PATH.py*) was developed to calculate travel time for the given flight routes. A high-level summary of the implementation is presented in Algorithm 4.

Algorithm 4 Travel Time Function in *PATH.py*

Implement the Lambert Conformal Conic map projection
Specify the user-defined maximum distance (d)
function ESTIMATED TRAVEL TIME(Origin, Destination, True Airspeed, Time)
    Calculate origin-destination travel distance (D) using the Haversine formula
    if $D < d$ **then**
        Compute aircraft initial bearing
        Estimate wind values using the *WIND.py* module [103]
        Calculate aircraft ground speed (G)
        travel time = D / G
    else if $D > d$ **then**
        Discretize the trajectory into N segments
        for $segment = 1, 2, \ldots, N$ **do**
            Calculate travel distance for the segment using the Haversine formula (D_s)
            Compute aircraft initial bearing at the start point of the segment
            Estimate wind values at the start point of the segment using the *WIND.py*
            Calculate aircraft ground speed at the start point of the segment (G_s)
            travel time for the segment = D_s / G_s
        end for
        Sum travel time for all the segments
    end if
end function

A straight line is typically considered as the shortest distance in the Cartesian coordinate and the distance is calculated by the Euclidean distance formula. Conceptually, the straight line is no longer the shortest distance on the surface of the Earth; but instead, the great circle distance is normally recognized as the shortest distance between start and end points on the Earth. There are various algorithms that compute the great circle distance, which mainly depends on a specific shape. For example, the Haversine formula assumes that the shape is a sphere while the Vincenty formula is an ellipse. In general, the Haversine formula is a simpler computation for calculating the great circle distance; however, it does not always provide the accuracy that the Vincenty formula offers. The Vincenty formula may be more accurate than the Haversine formula; however, it is more computationally intensive. Since this research was time-sensitive, the Haversine formula was utilized to calculate the great circle distance between two points on the Earth. In particular, the flight trajectory between two points was discretized to account for the curvature of the Earth when the two points were distant. The Haversine formula for calculating the great circle distance (h) is defined in Equation 5.3 [104].

$$h = 2R \arcsin(\sqrt{\sin^2(\frac{\psi_2 - \psi_1}{2}) + \cos(\psi_1)\cos(\psi_2)\sin^2(\frac{\lambda_2 - \lambda_1}{2})}), \tag{5.3}$$

where R is the radius of Earth, ψ represents latitude of the given point in radian, and λ represents longitude of the given point in radian.

Aircraft ground speed can be determined by the vector sum of the aircraft true airspeed and wind speed. The true airspeed information was retrieved from the ANSP. The eastward/northward wind speed was estimated by Enabler 1 [103] that combines a supervised machine learning-based wind regression model with the IDW technique. The formula for calculating the initial bearing of aircraft is defined in Equation 5.4 [104]. It should be noted that the initial bearing (θ) is assumed to be constant on the discretized segment.

$$\theta = \operatorname{atan2}(\sin\Delta\lambda \cdot \cos\psi_2, \cos\psi_1 \cdot \sin\psi_2 - \sin\psi_1 \cdot \cos\psi_2 \cdot \cos\Delta\lambda), \tag{5.4}$$

where ψ_1 is the latitude of the start point, ψ_2 is the latitude of the end point, and $\Delta\lambda$ is the difference in longitude of the end point and the start point. Figure 5.33 illustrates how aircraft ground speed is computed by aircraft true airspeed, aircraft initial bearing, and estimated winds. With aircraft ground speed and distance for the segment, total travel time can be calculated by integrating all of the segments.

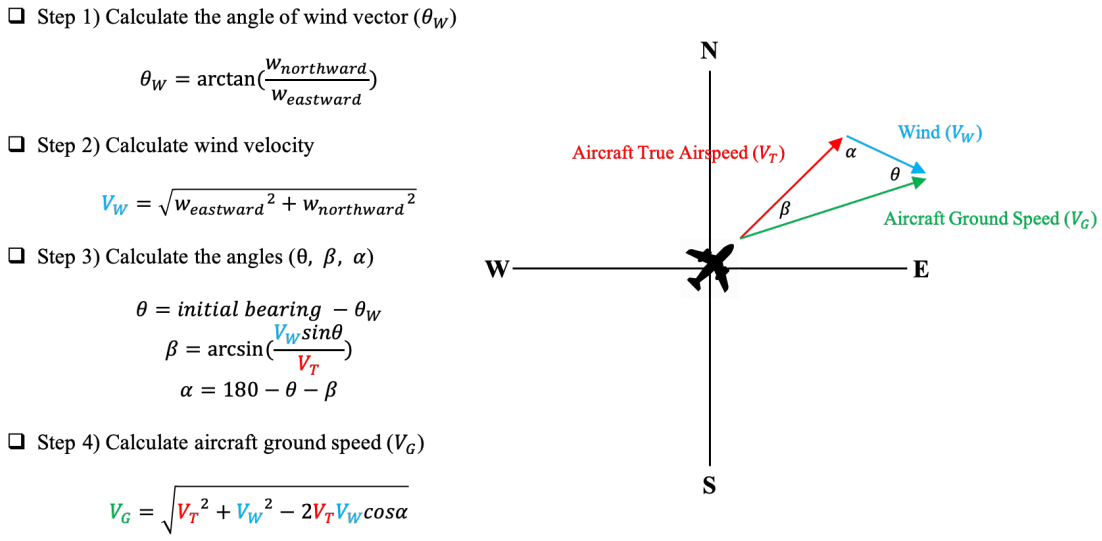


Figure 5.33: Steps for calculating of aircraft ground speed

5.4.5 Collision Check

In addition to the travel time function, the Python code (i.e., *PATH.py*) includes a collision check function to check a collision between aircraft and obstacle polygons. A high-level summary of the implementation is presented in Algorithm 5. Figure 5.34 notionally illustrates how the collision check in Algorithm 5 is performed.

Once Enabler 2 [35] generates new polygons, a collision check is conducted using the Shapely Python library [105]. The collision check is performed for all the polygons that exist at the time. Here, the polygons represent hazardous convective weather activity designed to give high penalty costs in a way that the weight factor is multiplied to travel

Algorithm 5 Collision Check Function in *PATH.py*

Implement the Lambert Conformal Conic map projection
Generate N polygons using the *SIGMET.py* module [35]
Specify the user-defined weight parameter W
function CHECK COLLISION(Start Point, End Point, True Airspeed, Time, Polygon Information)
    Decompose aircraft true airspeed into velocity vector components
    for $polygon = 1, 2, \ldots, N$ **do**
        if polygon movement speed = 0 **then**
            StaticLine = shapely.geometry.LinString(start point, end point)
            if StaticLine.intersects(polygon[i]) = True **then**
                there is a collision
                cost = W * cost
            else if StaticLine.intersects(polygon[i] = False **then**
                there is no collision
                cost = cost
            end if
        else if polygon movement speed != 0 **then**
            Decompose polygon moving speed into velocity vector components
            Convert longitude/latitude into x/y coordinates using the Lambert Conformal Conic map projection
            Subtract the polygon velocity to both aircraft and polygon velocities
            Specify an imaginary x/y points by considering the relative velocity
            Convert the imaginary x/y coordinates into longitude/latitude coordinates
            DynamicLine = shapely.geometry.LinString(start point, imaginary point)
            if DynamicLine.intersects(polygon[i]) = True **then**
                there is a collision
                cost = W * cost
            else if DynamicLine.intersects(polygon[i] = False **then**
                there is no collision
                cost = cost
            end if
        end if
    end for
end function

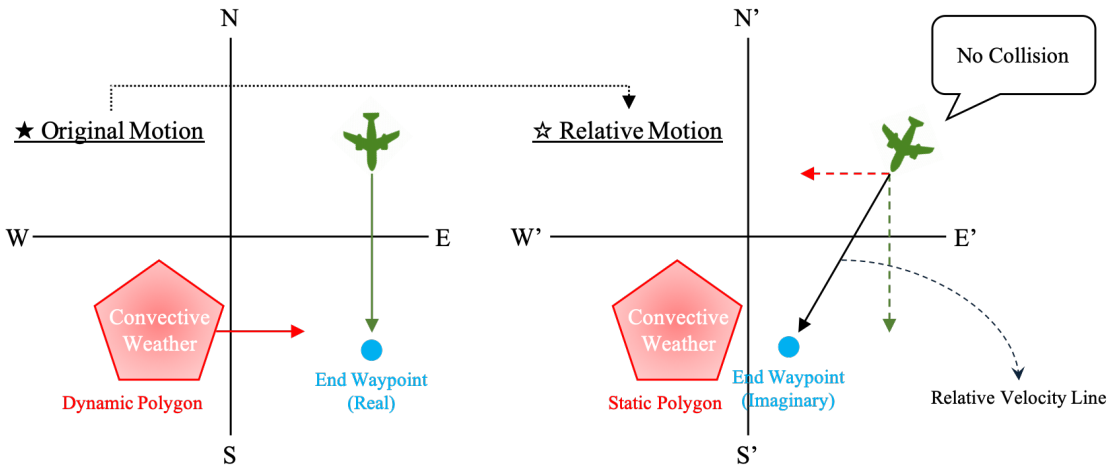


Figure 5.34: Notional sketch of how the collision check function works

time if aircraft penetrates them in a simulation. In reality, TFRs are also considered as an obstacle where pilots must avoid them during their flights; however, the TFRs are not considered in this research because they are typically lower than usual commercial aircraft en-route altitudes. To create the imaginary points by considering the relative velocity, the Lambert Conformal Conic (LCC) [106], which is a standard projection method for mapping large areas, was utilized to convert between longitude/latitude coordinates in the sphere and x/y coordinates in the plane. The transformation is performed with the following equations [107]:

$$x = \rho sin[n(\lambda - \lambda_0] \tag{5.5}$$

$$y = \rho_0 - \rho cos[n(\lambda - \lambda_0] \tag{5.6}$$

$$n = \frac{ln(cos\varphi_1 sec\varphi_2)}{ln[tan(\frac{\pi}{4} + \frac{\varphi_2}{2})cot(\frac{\pi}{4} + \frac{\varphi_1}{2})]} \tag{5.7}$$

$$F = \frac{cos\varphi_1 tan^n(\frac{\varphi_1}{2})}{n} \tag{5.8}$$

$$\rho = RFcot^n(\frac{\pi}{4} + \frac{\varphi}{2}) \tag{5.9}$$

$$\rho_0 = RFcot^n(\frac{\pi}{4} + \frac{\varphi_0}{2}), \tag{5.10}$$

where λ is the longitude, φ is the latitude, λ_0 is the reference longitude, φ_0 is the reference latitude, φ_1 and φ_2 are the standard parallels (e.g., φ_1 is the first standard parallel for lambert conformal), and R represents the radius of the Earth.

In fact, there have been many attempts to develop various map projection approaches (e.g., cylindrical, conic, and equirectangular projection) that are generally defined as a way that flattens the Earth surface into a two-dimensional plane. This is because the methods are always distorted in different ways such that there is no one particular map projection approach that works best for everything.

In this research, the LCC projection approach, which is one of the conic map projection methods introduced by Johann Heinrich Lambert in 1772, was selected for the following reasons: 1) it is common in many maps for the U.S., 2) it is commonly used in the aviation industry because a straight line drawn on the LCC projection map is almost similar to a great circle distance line, and 3) the U.S. Visual Flight Rules (VFR) charts are typically drafted on the LCC projection map.

5.4.6 Route Generation

A Search Algorithm*

The A* search algorithm is the core capability of the designated points-based flight path optimization model proposed in this research. There are numerous ways to implement the A* search algorithm to perform flight path planning with the aim of minimizing travel time. This research specifically implemented the tailored A* search algorithm that performed the search on an irregular graph that was made of the architecture of U.S. airspace. A high-

level summary of the implementation is presented in Algorithm 6. Particularly, the Heap data structure, which is also known as priority queue that satisfies the heap property [108], was used to efficiently handle prioritization in a list.

Ball Tree

In addition to creating a bounding box, the Ball Tree is another feature implemented in the Python code (i.e., *PATH.py*) to handle potential issues associated with computational complexity. This feature was designed to reduce computational costs especially when the designated points-based flight path optimization model needed to find the nearest neighbor waypoints more efficiently.

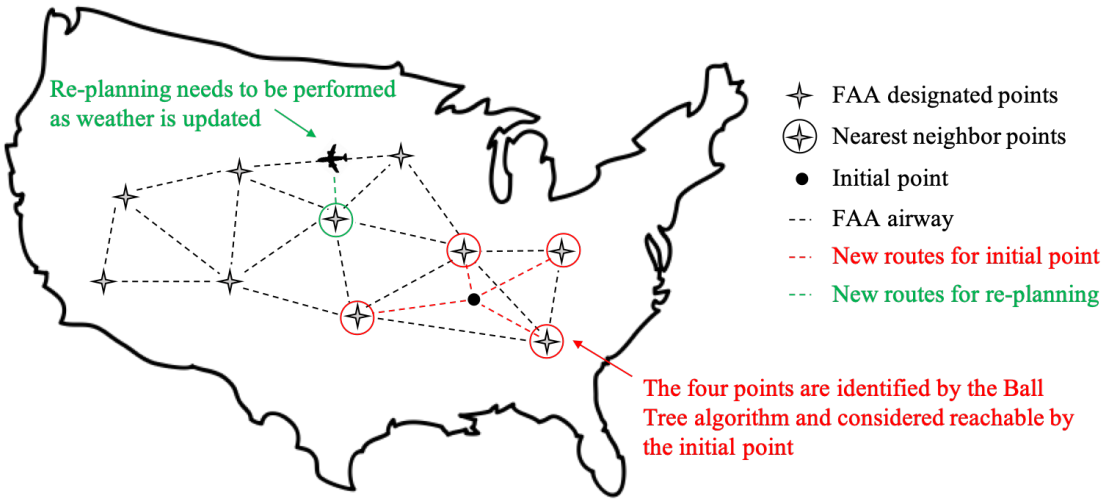


Figure 5.35: Notional sketch of the necessity for the Ball Tree implementation

This research particularly utilized the Ball Tree for the following reasons: 1) there are no routes connected to the initial and final point of a flight trajectory, and 2) there are no routes connected to the point where in-flight re-planning is performed after weather information is updated. For these cases, the nearest neighbor waypoints were considered reachable by the current aircraft position points. The Haversine formula was used to calculate the distance between the waypoints. Figure 5.35 notionally illustrates the necessity

Algorithm 6 Path Optimization Function in *PATH.py*

function PATH REPLANNING(Origin, Destination, True Airspeed, Time, Polygons, U.S. Airspace Infrastructure)
    Establish U.S. airspace infrastructure within a bounding box
    Generate N polygons using the *SIGMET.py* module [35]
    Evaluate cost for the start point
    Update close list (C) with the start point's cost and coordinate
    Find waypoints connected to the start point
    Update open list (O) with the connected point's costs and coordinates
    Heapify O
    Update C with the minimum cost case (i.e., current point)
    while O is not empty AND the current point in C is not destination **do**
        Find waypoints connected to the current point
        for $waypoint = 1, 2, \ldots, N$ **do**
            if waypoint[i] is already in C **then**
                Skip the waypoint[i]
            else if waypoint[i] is not in C **then**
                if waypoint[i] is already in O **then**
                    if new cost is lower than the previous cost **then**
                        Remove the previous case
                        Evaluate cost for the waypoint[i] and check collision
                        Add the waypoint[i] to O
                        Heapify O
                    else if new cost is larger than the previous cost **then**
                        Skip the waypoint[i]
                    end if
                else if waypoint[i] is not in O **then**
                    Evaluate cost for the waypoint[i] and check collision
                    Update O with the waypoint[i]
                    Heapify O
                end if
            end if
        end for
        Update C with the minimum cost case (i.e., current point)
    end while
    Identify the optimal flight trajectory in C by referring parent points
    Reverse the flight trajectory
end function
Repeat the function as weather information is updated

of the Ball Tree implementation for Enabler 3 (i.e., designated points-based flight path optimization model).

Additional Free-Flight Route Generation

To generate additional free-flight routes that were designed to relax the pre-determined U.S. airspace constraints, the route generation function was created in the Python code (i.e., *PATH.py*). Figure 5.36 notionally describes how the function generates additional free-flight routes as needed.

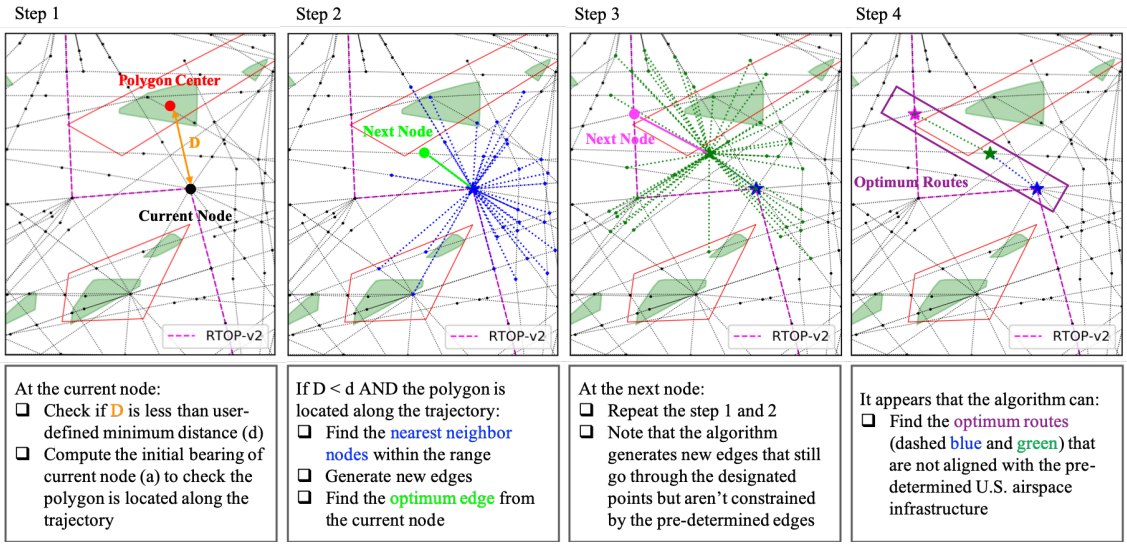


Figure 5.36: Overview of the additional free-flight route generation process

More specifically, at the current node in Step 1, the algorithm checks if additional free-flight route generation is needed by calculating the distance (D) between the polygon's centers and the current node. Here, it is important to note that the algorithm only considers the polygons that are located along the flight trajectory as described in Figure 5.37. If it identifies its necessity (i.e., D is less than the user-defined minimum distance and there is a polygon along a flight trajectory), it finds the nearest neighbor points within the user-defined range by using the Ball Tree algorithm (e.g., blue points in Step 2). The points

are considered reachable by the current node. The algorithm generates additional free-flight routes based on the points (e.g., dashed green lines in Step 3) and removes some of the routes if there are duplicates compared to the FAA pre-determined air routes (e.g., dashed black lines). The algorithm finds the optimal trajectory from the current node and repeats the process until it reaches the destination. As can be seen in Step 4, it appears that the algorithm generates two additional free-flight routes that are not aligned with the FAA pre-determined air routes. It should be noted that the algorithm generates the additional free-flight routes only when the following conditions are satisfied to save computational costs: 1) the distance between the current node and the polygon center is less than the user-defined minimum distance and 2) the polygon is located along the trajectory.

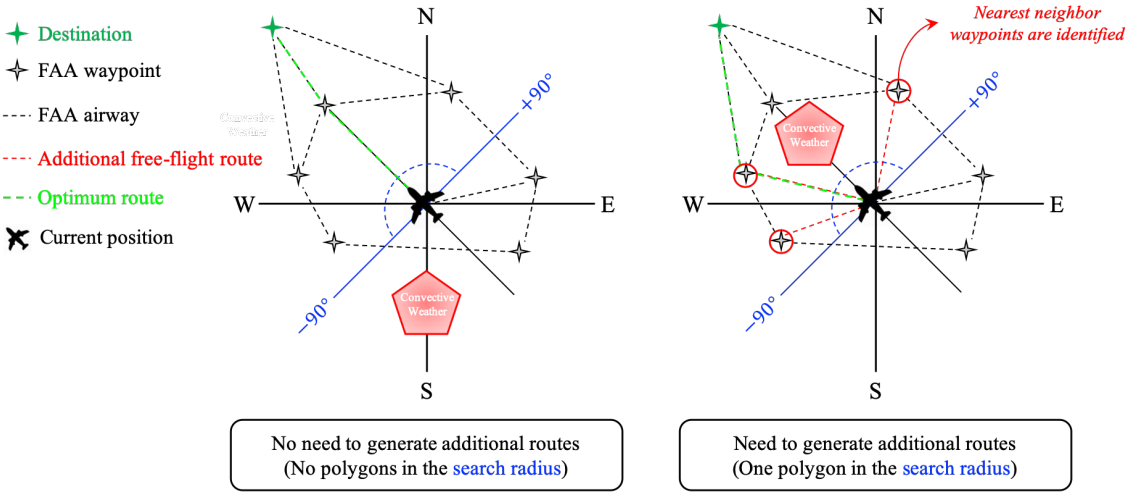


Figure 5.37: Notional sketch of the necessity to generate additional free-flight routes

In summary, Figure 5.38 shows a notional process of the proposed methodology that implements three different approaches with primary datasets: 1) a supervised machine learning-based wind prediction model (i.e., Enabler 1) to obtain continuous wind information, 2) an unsupervised machine learning-based short-term convective weather model (i.e., Enabler 2) to delineate reliable and up-to-date areas of convective weather, and 3) a designated points-based flight path optimization model (i.e., Enabler 3) that combines the

A* search algorithm with the free-flight approach to optimize flight routes that are not necessarily constrained by the pre-determined U.S. airspace infrastructure but provide more acceptable free-flight routes.

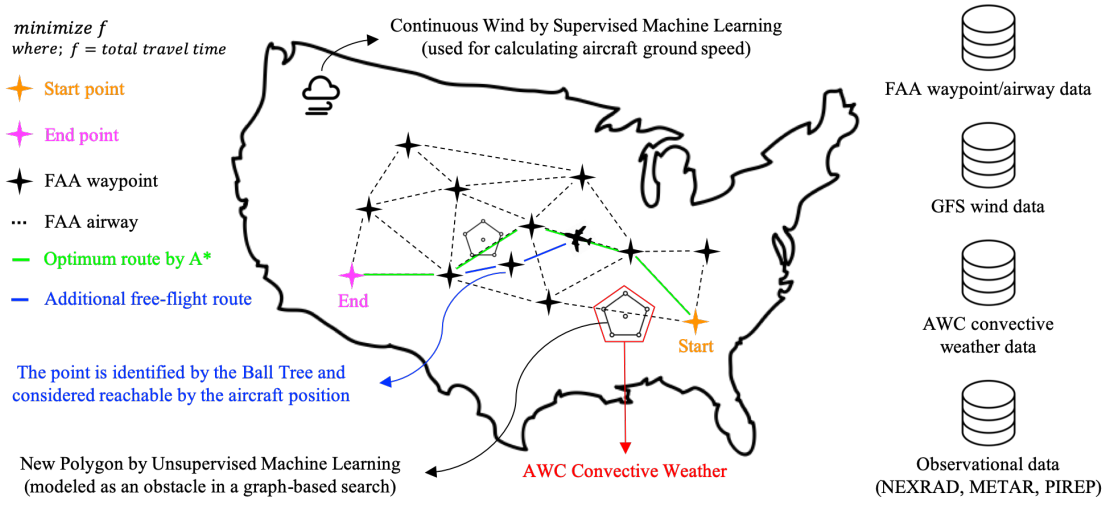


Figure 5.38: Notional process of the proposed methodology

5.4.7 Research Experiment

To satisfy the second requirement (i.e., the capability of providing simulated optimum flight routes automatically to a cockpit-based real-time flight path optimization framework) raised by the Research Question 2, this research specifically constructed the Research Hypothesis 2.1 as follows:

> *Research Hypothesis 2.1: A hybrid method that combines the A* search algorithm with a free-flight approach will automatically provide flight route opportunities that are not necessarily constrained to established waypoint-to-waypoint airways but rather generate acceptable free-flight route options.*

This section introduces the Research Experiment 2.1 to investigate and test the Research Hypothesis 2.1. The purpose of this research experiment is to see how much a simulated

optimum flight trajectory generated by Enabler 3 is similar to a real flight trajectory, especially under hazardous weather conditions. The procedure implemented for this research experiment is described as follows: Step 1 is to select the date where hazardous convective weather activities are dominant and collect corresponding historical weather data for the selected date. Step 2 is to generate two optimal flight routes with different options: 1) use a designated points-based flight path optimization model (i.e., hybrid method combining the A* search algorithm with a free-flight approach) and 2) find an optimal flight route using only the A* search algorithm. Step 3 is to compute the Hausdorff distance for the two scenarios to compare them with a real flight trajectory created by flight dispatchers of major airlines.

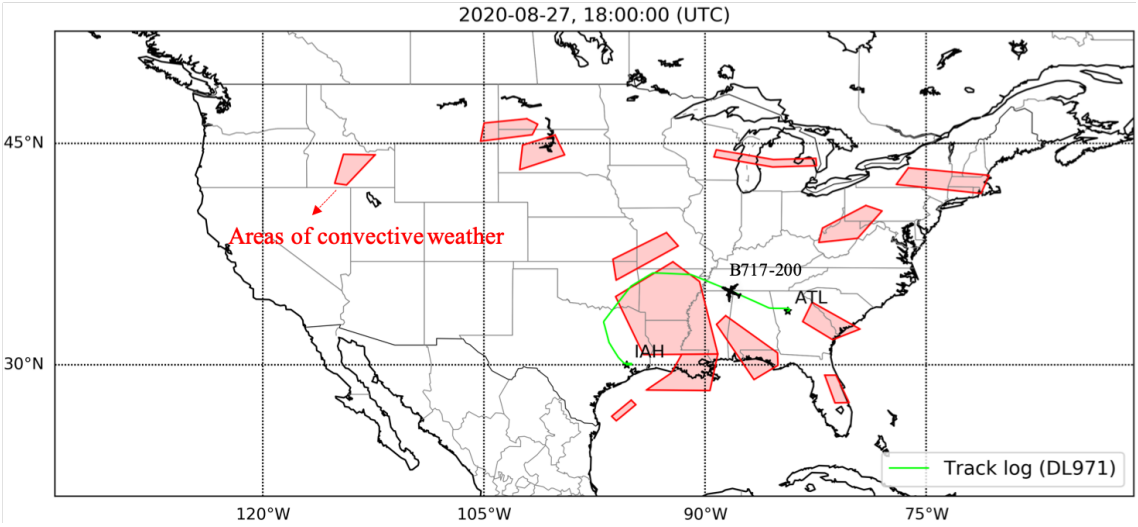


Figure 5.39: DL971 flight path visualization

As a part of this research experiment, one case study was performed with the previous Delta Airlines Flight 971 as a proof-of-concept. The intent of this case study was to compare the simulated flight trajectory generated by Enabler 3 (i.e., designated points-based flight path optimization model) with the actual flight (i.e., DL971) trajectory especially when severe weather was present. Given that most flights were either canceled or delayed

on August 27th, 2020 as hurricane Laura hit Louisiana, the date was intentionally selected to see the impact of hurricane Laura (i.e., formed on August 20th, 2021 and dissipated on August 29th, 2021) on commercial airline's flight trajectory in U.S. airspace. Figure 5.39 shows the trajectory of the previous DL971 Flight used for this case study and Figure 5.40 visualizes the radar intensity of hurricane Laura at the time.

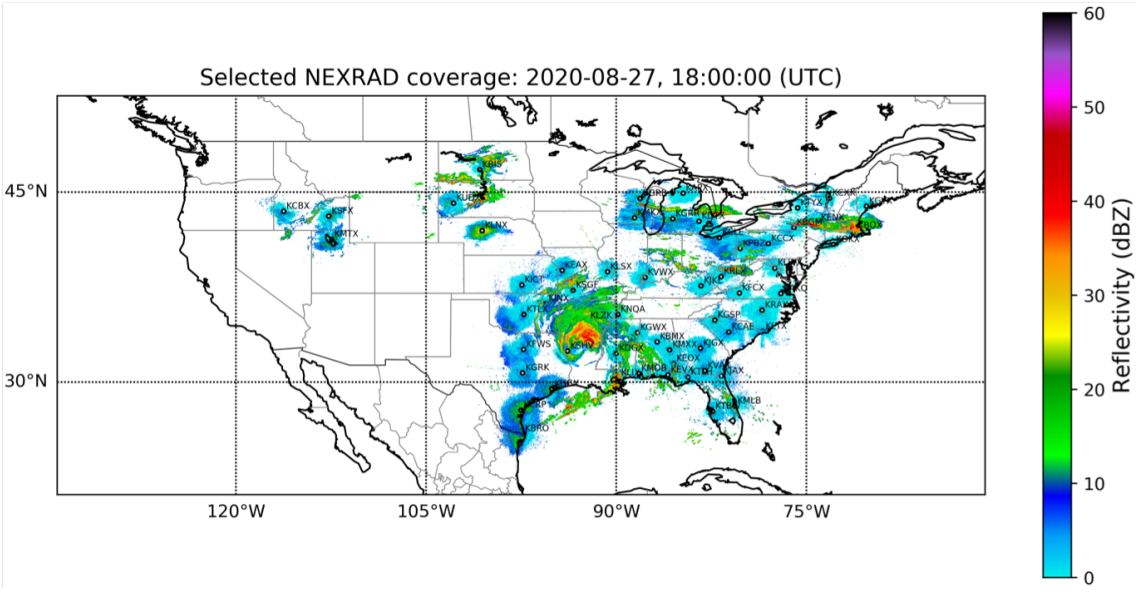


Figure 5.40: Radar intensity of hurricane Laura at 2020-08-27 18:00 UTC

Two different metrics (i.e., travel time and Hausdorff distance) were used to compare the simulated flight trajectory with the actual flight trajectory under hurricane Laura. As mentioned in Chapter 4, one way to understand the Hausdorff distance in this case study is to think of it as a metric that determines how a flight trajectory generated by simulation is similar to a real flight trajectory. Intuitively, the smaller the Hausdorff distance is, the more similar the two trajectories are.

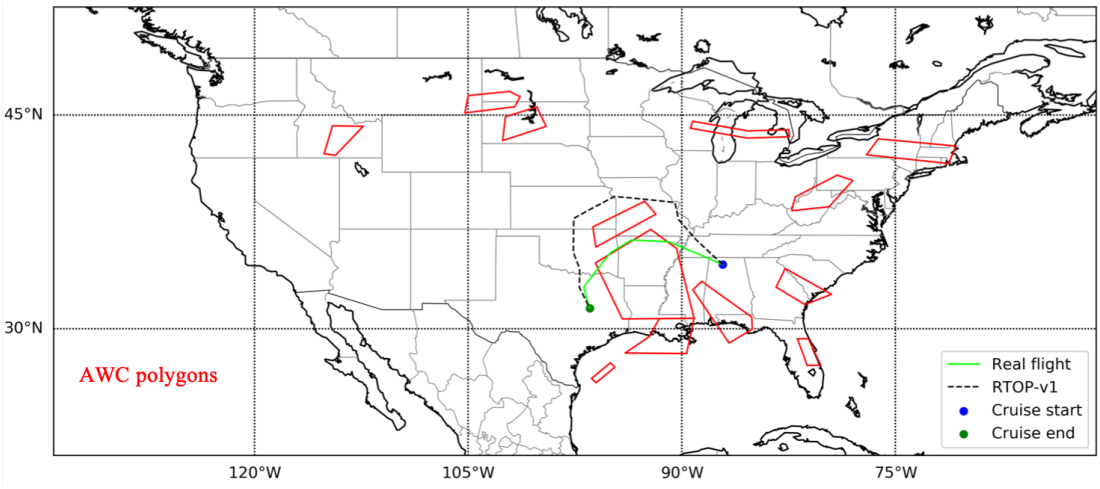
A comprehensive validation with several ASDL-initiated flight path optimization tools summarized in Table 2.1 was performed to demonstrate the applicability of the proposed methodology. A final visualization of the comparison between the actual flight trajectory

and the optimized flight trajectories generated by the ASDL-initiated tools is shown in Figure 5.41. It is important to note that 1) the PARTNER was not considered in the validation process as the PARTNER did not incorporate weather forecast information into the framework, 2) the RTOP-v1 used a linear interpolation method to yield continuous wind information, and 3) both RTOP-v2 and RTOP-v3 used Enabler 1 (i.e., supervised machine learning-based wind model) to obtain continuous wind information.

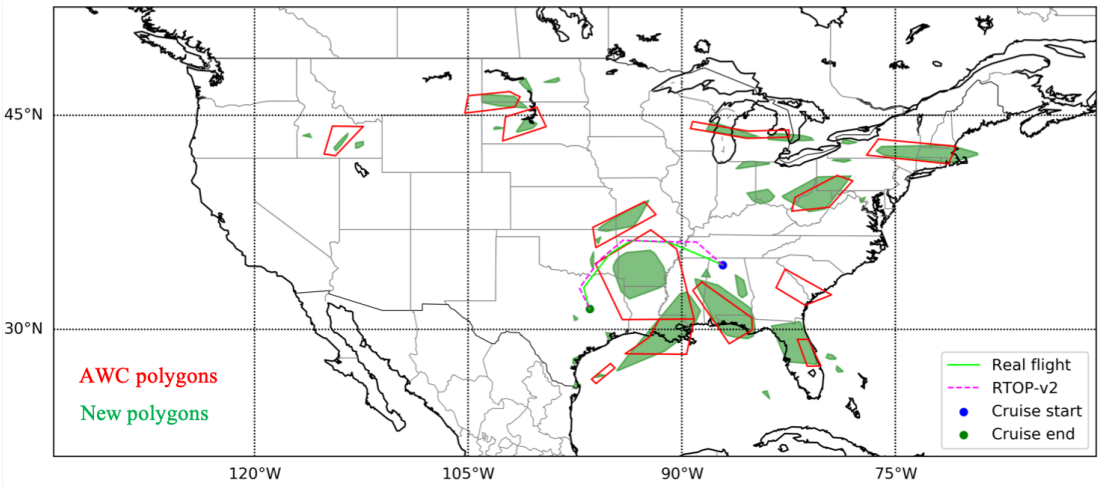
The results in Figure 5.41 indicate several important observations as follows: First, the RTOP-v1 generated the simulated flight trajectory where aircraft flew unnecessary detours due to the limitation of the AWC convective SIGMET data [35]. In other words, since the RTOP-v1 considered the red polygons issued by the AWC as a hard constraint, it generated such a big detour on the flight route. Second, the RTOP-v2 generated the simulated flight trajectory acceptable compared to the trajectory generated by the RTOP-v1 because the RTOP-v2 considered the green polygons generated by Enabler 2 (i.e., unsupervised machine learning-based short-term convective weather model) as a hard constraint. However, the flight trajectory generated by the RTOP-v2 was slightly different compared to the real flight trajectory especially at the beginning of the flight route. This was because Enabler 3 (i.e., designated points-based flight path optimization model) was not implemented in the RTOP-v2; thus, the flight route generated by the RTOP-v2 was conservatively constrained by the pre-determined U.S. airspace infrastructure. Last, the RTOP-v3 generated the simulated flight trajectory that was the most similar compared to the real flight trajectory.

Table 5.5: Travel time and Hausdorff distance comparison between DL971 and RTOP

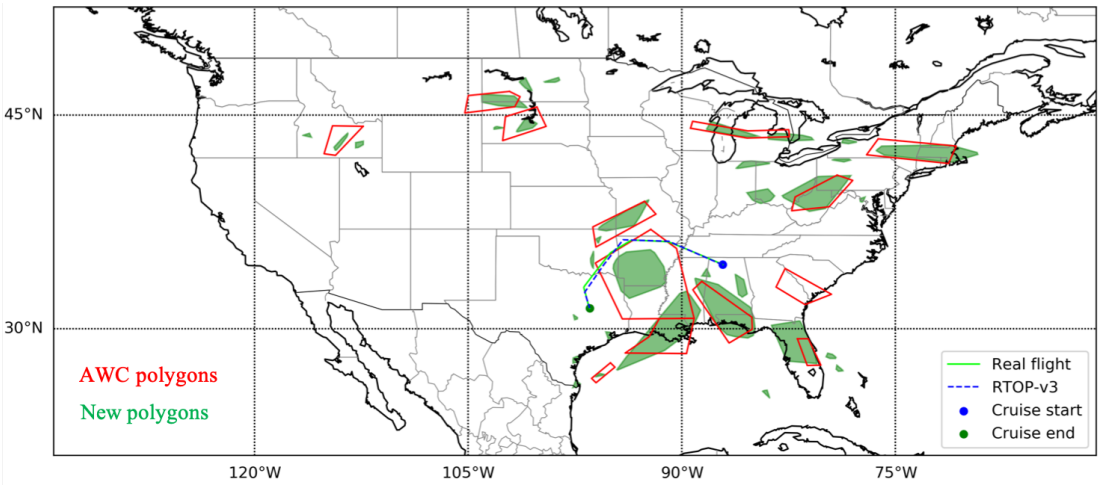
Method	Travel time (second)	Hausdorff distance
RTOP-v1	8508	3.53
RTOP-v2	5889	0.76
RTOP-v3	5565	0.31
DL971	5601	0.00



(a) Simulation results generated by the RTOP-v1



(b) Simulation results generated by the RTOP-v2



(c) Simulation results generated by the RTOP-v3

Figure 5.41: Simulation results generated by the RTOP

Table 5.5 shows the results of the two metrics (i.e., travel time and Hausdorff distance) used for the comparison between the actual flight (i.e., DL971) and the simulated flight cases depending on the version of the RTOP. This result implies the following significant observations: First, the results show that the RTOP-v3 provided the lowest Hausdorff distance compared to the other RTOP versions, indicating that the RTOP-v3 would accurately simulate the behavior of the actual flight trajectory operated by major airlines (e.g., Delta Airlines) in the U.S. under severe weather conditions. This is significant because it potentially means that the outcome of this research can be used to help small airline operators that do not necessarily have flight dispatchers but rather ask pilots to generate flight plans by providing the capability to proactively and continuously optimize flight routes under hazardous weather activity. Second, another important finding is that the RTOP-v3 even slightly reduced by approximately 35 seconds the duration of the en-route segment of the previous DL971 Flight case, implying that it has the potential to further reduce travel time by more than 35 seconds for other cases.

5.5 Software Architecture

The framework developed by this research is implemented in Python. This framework consists of a few modules with their primary datasets as shown in Figure 5.42. In a nutshell, the SIGMET module is designed to perform short-term convective weather predictions using an unsupervised machine learning algorithm. The WIND module facilitates a supervised machine learning algorithm to provide continuous wind information. The PATH module is designed to optimize aircraft trajectory by combining the A* search algorithm with a free-flight approach. In addition to the primary software components, the DATA module handles the download of aviation-related datasets and the DISPLAY module deals with data post-processing.

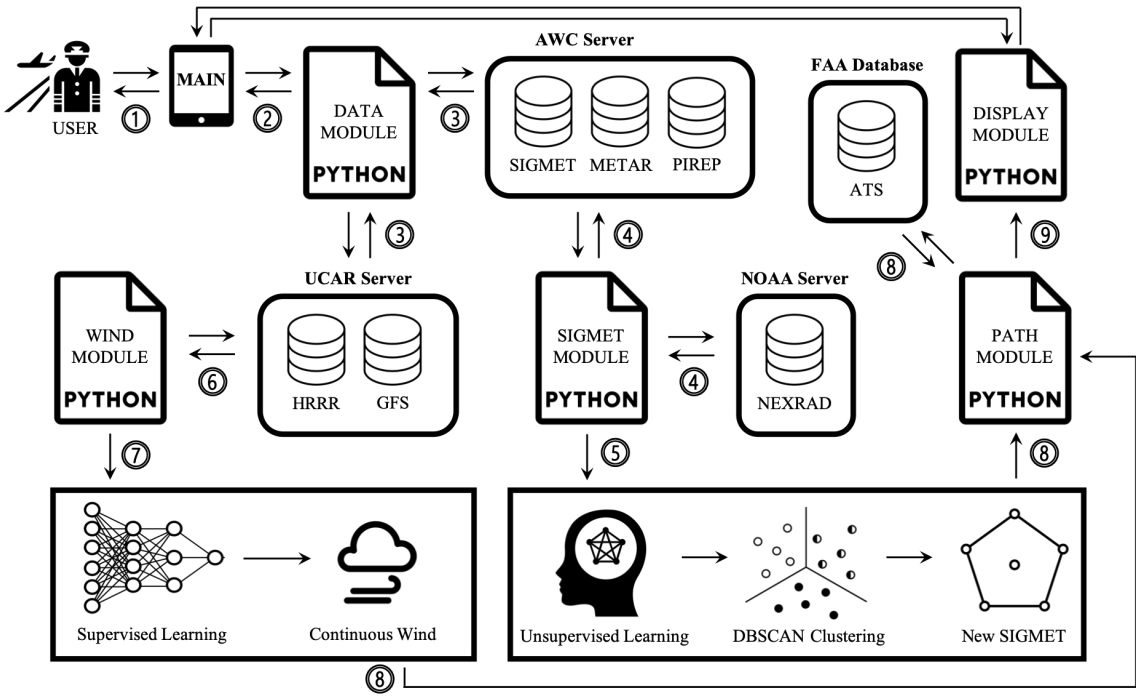


Figure 5.42: Software architecture used for this research

5.5.1 Data Pre-processing

The AWC publicly publishes textual and graphical weather forecasts to the aviation community with the aim of enhancing safe and efficient flights. Users can access aviation-related weather data such as METAR, SIGMET, and PIREP via the following link: https://www.aviationweather.gov/adds/dataserver_current/current/. Once downloading the aviation-related weather datasets at a specific time, the SIGMET module performs aviation-related data pre-processing for a filtering purpose. This filtering process is required for several reasons: 1) it significantly reduces processing time by filtering out METAR, PIREP, and NEXRAD that are not operationally relevant (i.e., it does not meet criteria defined by a user) and 2) it simplifies datasets so that it minimizes processing time in the other modules (e.g., minimizing radar processing time in the SIGMET module).

One potential barrier for those who want to keep track of accessing historical data is that the AWC provides a limited online access service. For example, METAR reports

are only available up to 36 hours. In response to this potential concern, a Python code (i.e., data retrieval) is developed to automatically access and download historical weather data published by the NOAA. The Python code starts by connecting the TDS server and downloads aviation-related weather datasets at a specific time defined by a user. The code is particularly designed to download datasets every hour as the datasets are hourly updated. A high-level summary of the Python code is presented in Algorithm 7.

Algorithm 7 Data Retrieval in *DATA.py*

```
1: Specify input UTC time
2: Specify download time interval
3: function DOWNLOADER(Input UTC time, Time interval)
4:     while A user stops the downloading process do
5:         Specify URLs
6:         Download aircraft-reports data
7:         Download SIGMET data
8:         Download METAR data
9:         Download PIREP data
10:        Download TAFs data
11:        Specify current UTC time
12:        Time sleep with time interval information
13:    end while
14: end function
```

In addition to accessing aviation-weather datasets, it is also important to obtain wind information in this research. Users can access wind datasets published by the NOAA via the following link: http://soostrc.comet.ucar.edu/data/grib/. Once downloading the wind datasets at a specific time, the WIND module performs data pre-processing steps primarily for decoding original data and preparing data for a machine learning training process.

5.5.2 Data Post-processing

The primary objective of the development of *DISPLAY.py* is to provide useful information for the pilot's decision-making process. This module includes the following functionalities: 1) waypoint visualization, 2) established waypoint-to-waypoint airway visualization, 3) the AWC convective SIGMET visualization, 4) the AWC PIREP visualization, 5) the

AWC METAR visualization, 6) the NOAA wind visualization, 7) the NOAA NEXRAD visualization, 8) optimized flight trajectory visualization, and 9) new convective weather polygon visualization. Based on the research scope, it is important to note that all these visualization functionalities are only valid within U.S. airspace.

While the ultimate goal of the DISPLAY module is to visually provide optimal flight routes to pilots, it must be noted that the pilots are still responsible for communicating with ATC because the framework developed by this research is not intended to replace on-board devices. It is expected that pilots will use flight route recommendations provided by the framework developed in this research for making a tactical avoidance decision as needed.

CHAPTER 6
RESULTS AND DISCUSSION

This chapter is aimed at constructing a set of simulation scenarios and examining a series of results obtained through statistical analyses. It is important to note that the results presented throughout this chapter are based on a series of assumptions (e.g., focusing only on the en-route phase) specified in this research.

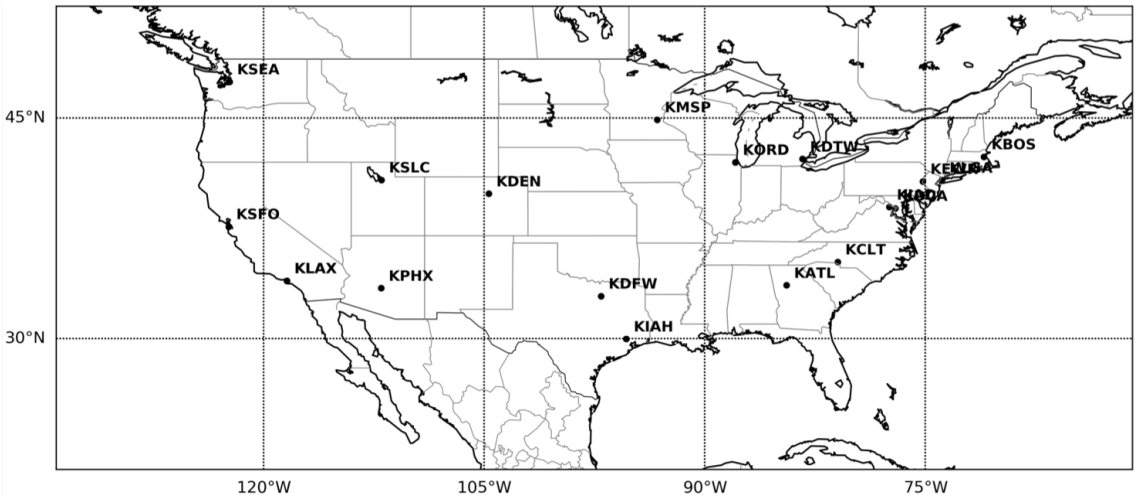
6.1 Statistical Analysis

Statistical analyses were performed using 491 real flights to not only highlight the statistical significance of simulation results but also get an indication of the savings to be expected.

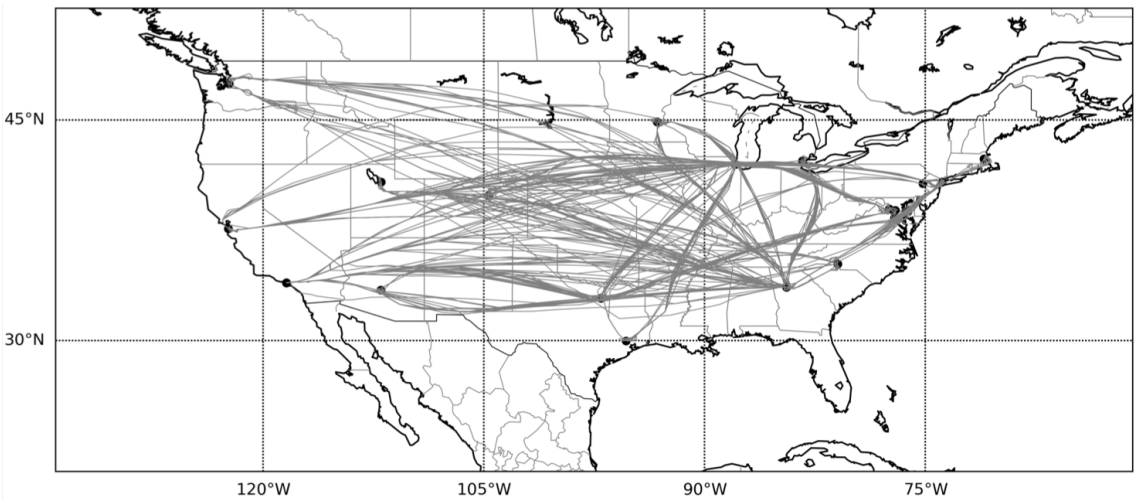
6.1.1 Flight Data Collection

Several aviation companies (e.g., FlightAware and Flightradar24) provide real-world flight tracking data that contains time, aircraft ground speed, altitude, latitude, longitude, and course direction with the origin, destination, airline, flight number, and aircraft identification. Given that FlightAware had sufficient real-world flight operation datasets received from the network of ADS-B multiple ground stations in many countries, some of the real-world flight operation records were retrieved from the FlightAware historical database for statistical analysis in this research.

The selected flight datasets contain one month of flight operations at the major airport hubs (e.g., ATL) of three major U.S. airlines (i.e., American Airlines, Delta Airlines, and United Airlines) with the most popular types of aircraft (e.g., B737-800 and A320 Neo) in use. This dataset consists of a total of 491 U.S. domestic flights. Figure 6.1 shows the selected airport hubs and historical flight trajectories used for statistical analysis in this research.



(a) Selected airport hubs



(b) Selected historical flight trajectories

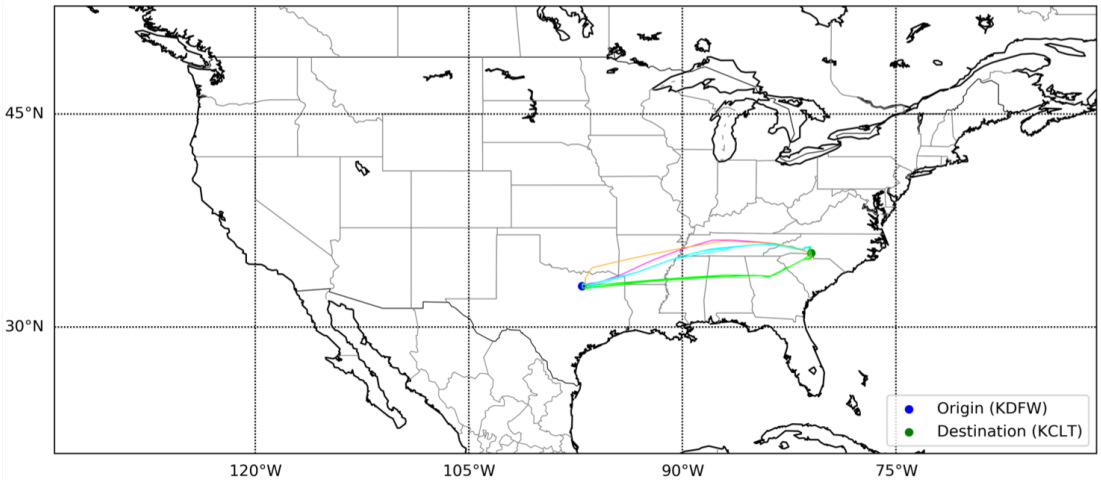
Figure 6.1: Selected flight datasets used for statistical analysis

6.1.2 Flight Trajectory Clustering

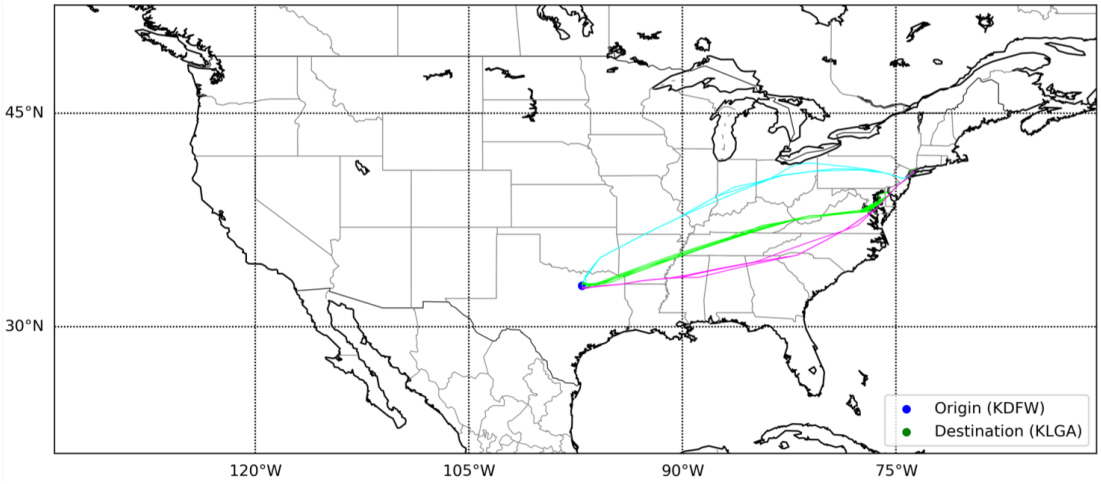
A trajectory clustering task is generally defined as a process partitioning a collection of trajectories into similar groups [109]. The trajectory clustering task, which is typically formulated as an unsupervised machine learning problem, has been widely used in various engineering areas such as civil and aerospace engineering. For example, Stefan Atev et al. [110] combined two spectral clustering methods to measure vehicle trajectory similarity. Andrew M. Churchill and Michael Bloem [111] proposed a method for clustering aircraft taxi trajectories to ultimately identify anomalous trajectories. In addition to the clustering of aircraft taxi trajectories, Samantha J. Corrado et al. [112] developed a data-driven anomaly detection method that clusters real-world flight trajectories to analyze terminal airspace operations.

Based on the fact that flight flows typically have varying densities, this research specifically utilized the Hierarchical Density Based Spatial Clustering of Applications with Noise (HDBSCAN) algorithm instead of the DBSCAN algorithm for clustering the selected historical flight trajectories. The intent of the HDBSCAN-based clustering task in this research is to identify the particular flight cases with two use-cases in mind: First, each cluster is reduced to a median representative flight case used for comparison with a simulated flight generated by the proposed methodology. Second, a noise case (i.e., anomalous trajectory in the group) identified by the HDBSCAN algorithm is used for comparison with a simulated flight generated by the proposed methodology. Such noise case is defined as a case where the flight trajectory is significantly different compared to the mainstream of the flight trajectories in the cluster group given that the noise cases are potentially due to the flight having had to make an excessive detour due to hazardous weather conditions.

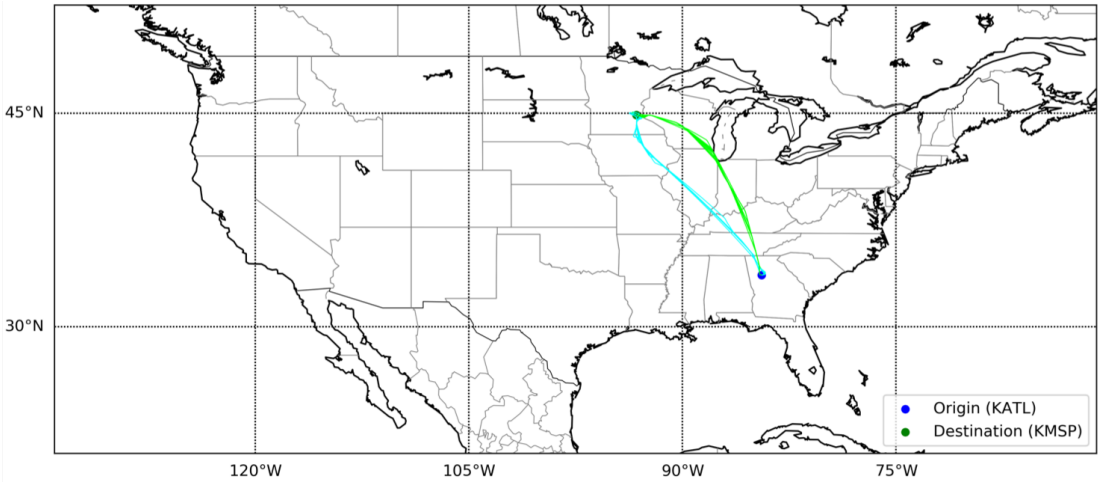
Figure 6.2 visualizes the resulting clusters for three different flight routes. It is important to note that the HDBSCAN algorithm used the Hausdorff distance-based square matrix containing the distances between pairs of each flight route to cluster the flight trajectories.



(a) Clustering results for DFW–CLT flights



(b) Clustering results for DFW–LGA flights

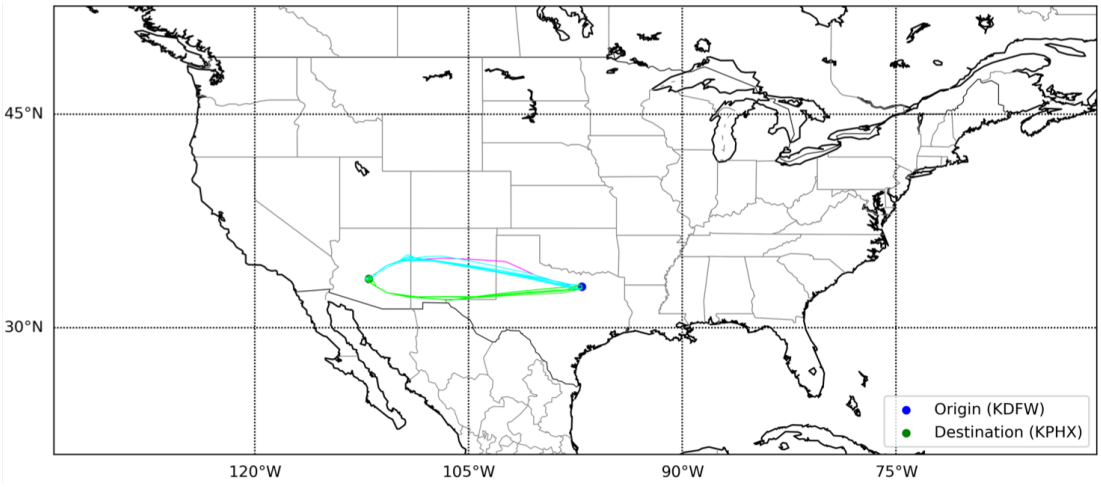


(c) Clustering results for ATL–MSP flights

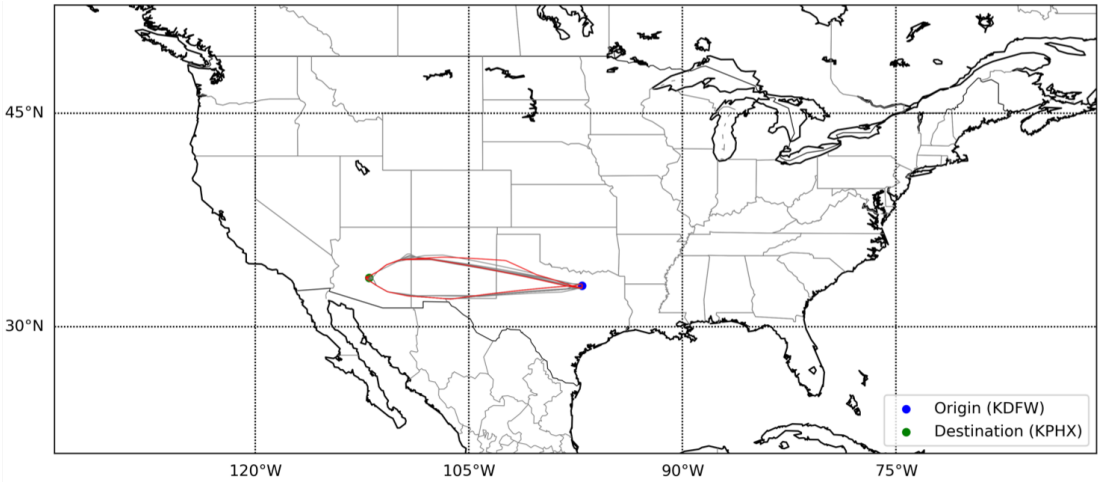
Figure 6.2: Clustering results for the selected historical flights

6.1.3 Representative Flight Selection

To compare an actual flight trajectory with a simulated flight trajectory generated by the proposed methodology, each cluster should be reduced to a single representative flight case. Given that the groups clustered by the HDBSCAN algorithm still contained a large number of flights, a median case of each cluster group was identified by computing the Hausdorff distance for all of the cases in the cluster group. For example, Figure 6.3 shows the results of flight trajectory clustering and median representation flight selection.



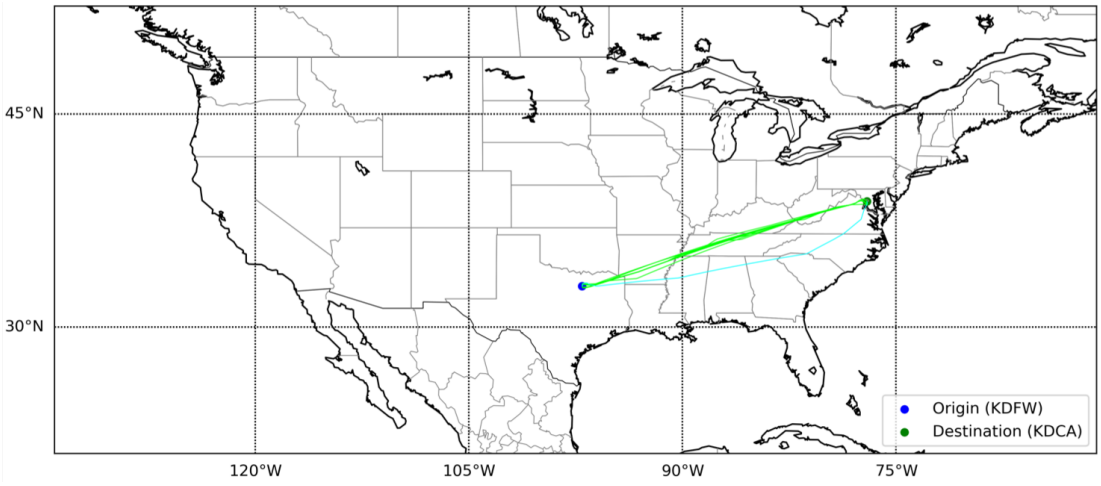
(a) Clustering results for DFW–PHX flights



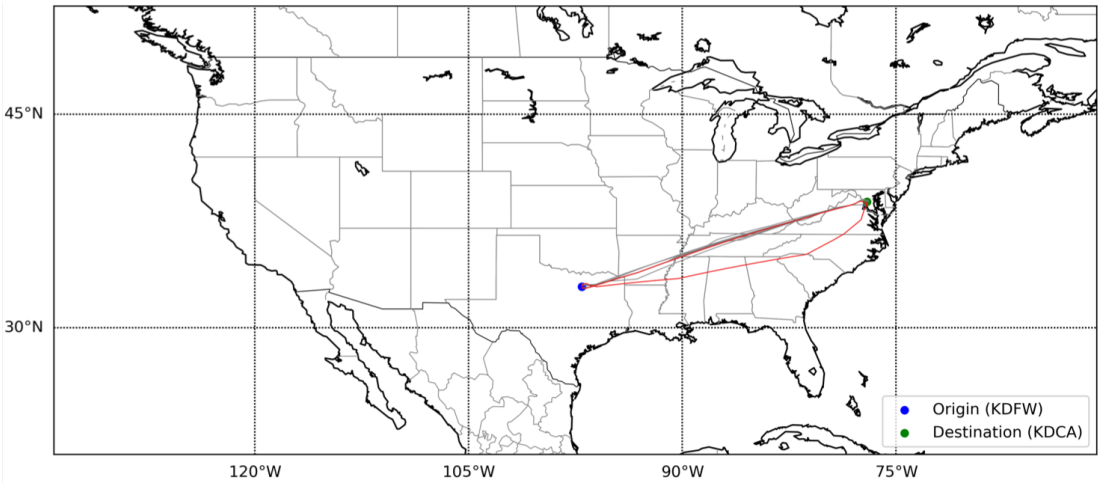
(b) Median representative flight results (i.e., red-colored flight trajectories) for DFW–PHX flights

Figure 6.3: Median representative flight selection for DFW-PHX flights

Noise cases identified by the HDBSCAN algorithm (i.e., anomalous flight trajectory in each cluster group) were also considered as representative flights given that the cases made a detour from the median flights primarily due to severe weather conditions. For example, Figure 6.4 shows the results of the flight trajectory clustering and one anomalous flight case (i.e., cyan-colored flight trajectory).



(a) Clustering results for DFW–DCA flights

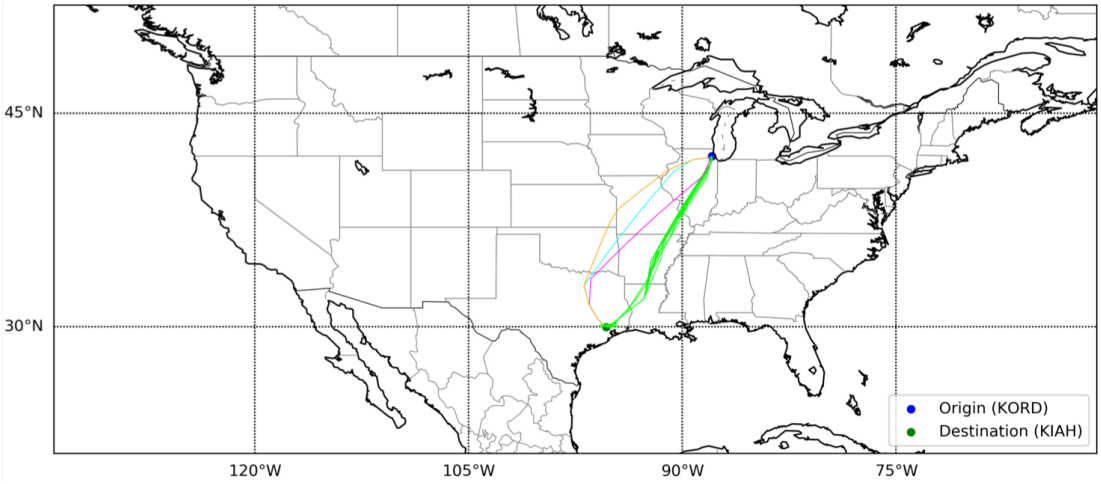


(b) Representative flight results (i.e., red-colored flight trajectories) for DFW–DCA flights

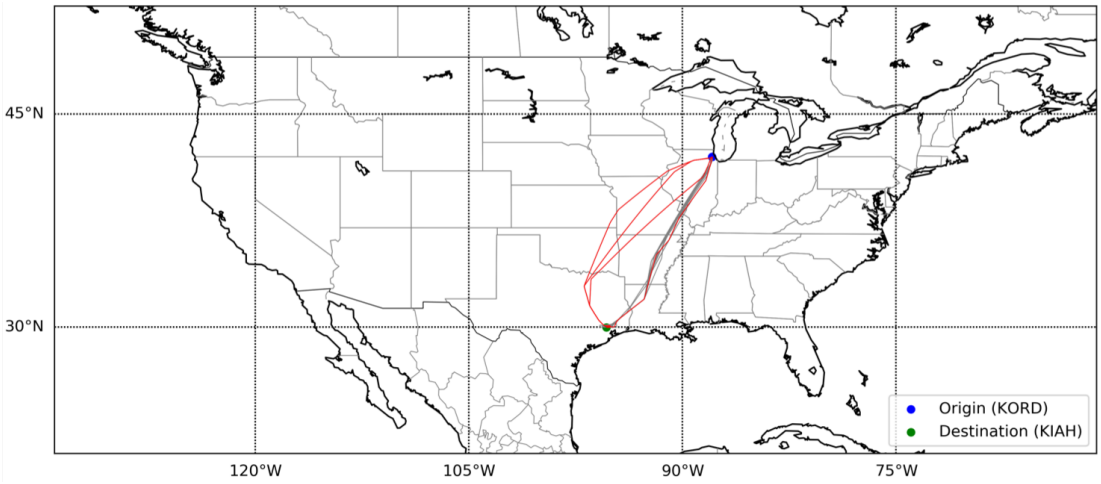
Figure 6.4: Representative flight selection for DFW-DCA flights

It is worth mentioning that the HDBSCAN algorithm sometimes identified more than one anomalous flight case for each cluster as the noise cases did not conform to an identified

representation of standard flight operations. For example, Figure 6.5 shows the results of the flight trajectory clustering and three anomalous flight cases (i.e., cyan/magenta/gold-colored flight trajectories).



(a) Clustering results for ORD–IAH flights



(b) Representative flight results (i.e., red-colored flight trajectories) for ORD–IAH flights

Figure 6.5: Representative flight selection for ORD-IAH flights

Real-world flight tracking datasets for the representative flights (i.e., median cases and anomalous cases) were retrieved from the historical FlightAware database. In addition to collecting real-world flight tracking data, the corresponding historical weather datasets (i.e., PIREP, SIGMET, METAR, NEXRAD, and GFS) for the representative flight cases were

also retrieved from different data servers (e.g., radar data from the Amazon web service, wind data from the NOAA, and convective weather data from the AWC) to construct a set of simulation scenarios comparing actual flights with simulated flights.

6.1.4 Verification and Validation

Before assessing the potential benefits of the framework developed in this research, an apples-to-apples comparison (i.e., it constrains the algorithm to follow the trajectory of real-world flights and calculates the corresponding duration of flights in the simulation environment) was conducted with the representative flights. More specifically, the algorithm went through the following steps as shown in Figure 6.6 to estimate the duration of the en-route phase in the simulation environment: 1) discretize the flight trajectory into N segments, 2) calculate the great circle distance for each segment, 3) compute aircraft ground speed for each segment using aircraft true airspeed information obtained from the ANSP and wind estimation predicted by the supervised machine learning-based wind model, and 4) accumulate travel time for all of the segments.

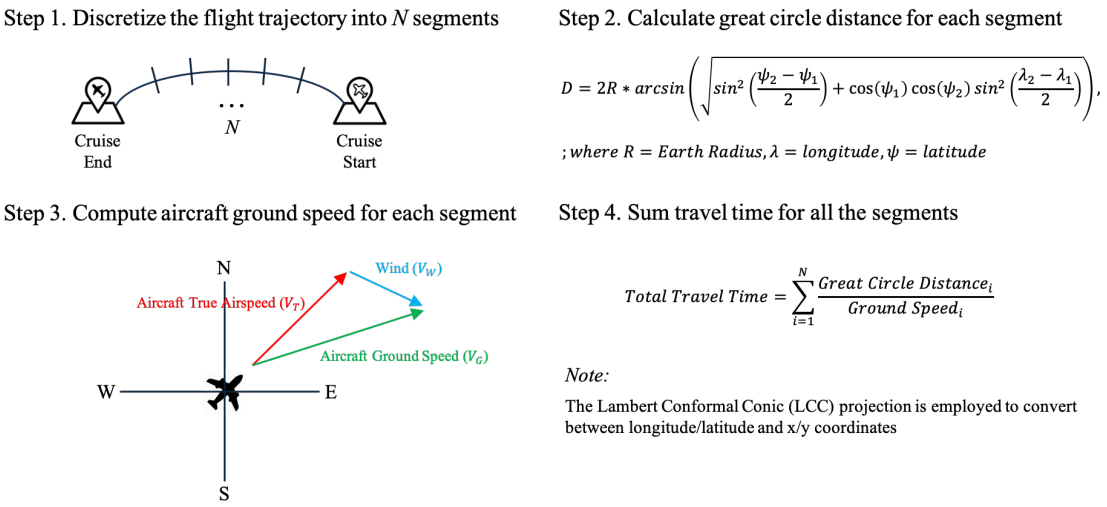
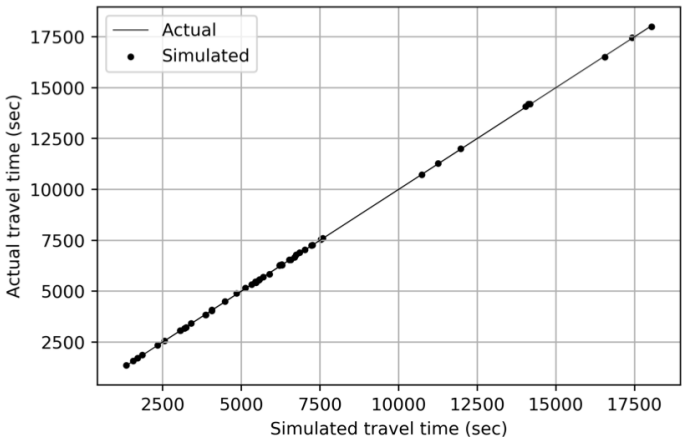
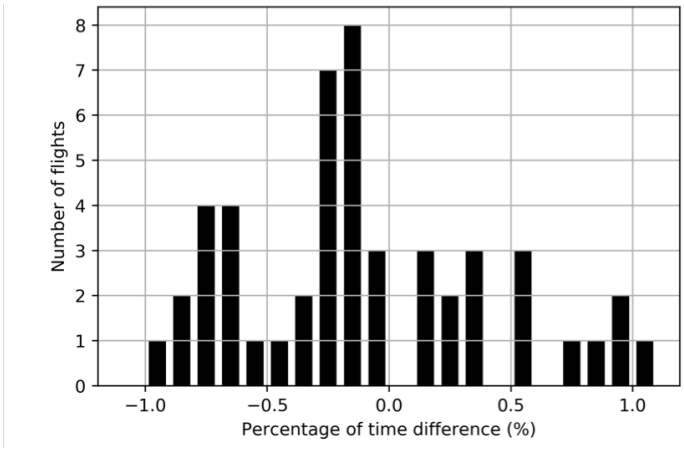


Figure 6.6: Steps for estimating travel time

Figure 6.7 shows the results of the comparison between actual flights and simulated flights for the representative flight cases. As can be seen, it shows the good accuracy of the simulation prediction, indicating that the framework developed in this research generates valid results as long as input data (e.g., aircraft true airspeed) is provided accurately.



(a) Actual vs. Predicted



(b) Percentage of time difference distribution

Figure 6.7: Apples-to-apples comparison between real flights and simulation cases

In particular, Figure 6.8 shows the trajectory of the median case of the representative flights. It is important to note that the comparison concentrated only on the en-route phase as shown in Figure 6.9. Since it appeared that the DL1944 flight flew with two different en-route phases, the algorithm constrained the simulated flight to follow the actual flight

trajectory with two different en-route phases (i.e., two different start and end points) corresponding to the two different flight altitudes.

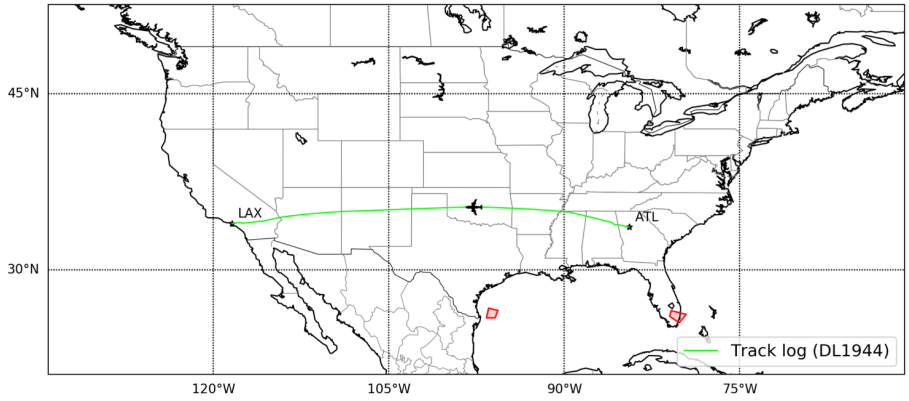


Figure 6.8: DL1944 flight path visualization

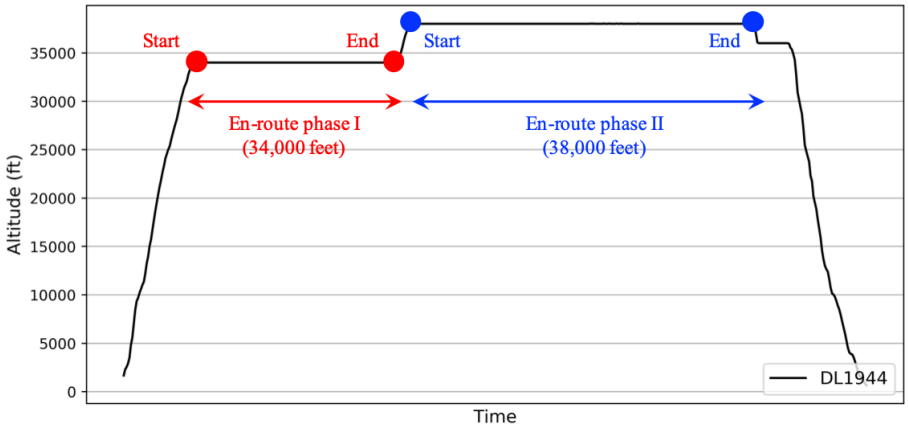


Figure 6.9: DL1944 mission profile

Comparing the duration of the en-route segment, there were only 16 seconds of difference between the real flight and the simulated flight, implying that the proposed methodology generates valid results as long as input data (e.g., aircraft true airspeed) is provided accurately. To prove the potential benefits of the framework, the simulation results of the duration of the en-route phase in Figure 6.7 were considered as a baseline used for comparing it with an optimal flight route generated by the proposed methodology.

6.1.5 Assessment of Potential Benefits

Given that the selected airlines (i.e., American Airlines, Delta Airlines, and United Airlines) generally hire flight dispatchers to proactively optimize flight routes while in-flight, this research specifically employed the Hausdorff distance as an evaluation metric that measures flight trajectory similarity (i.e., how much a simulated flight route is different from an actual flight route) between the real-world flight trajectories and the simulated flight trajectories optimized by the proposed methodology. Figure 6.10 shows the results of the Hausdorff distance for the representative flight cases.

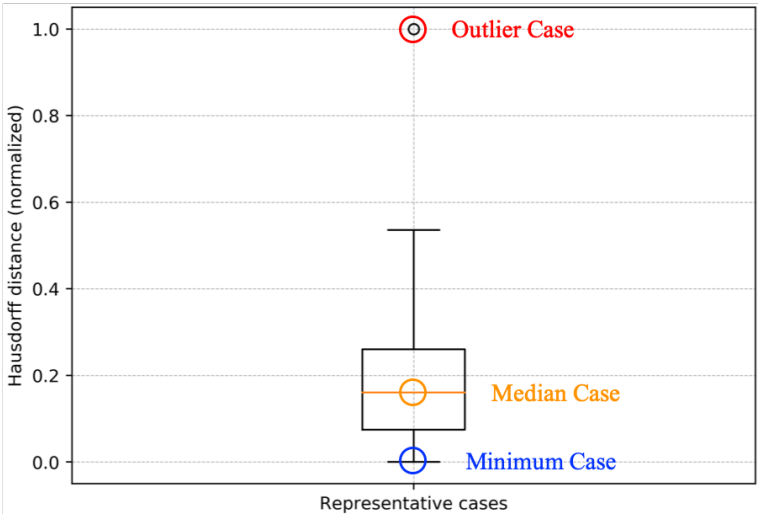
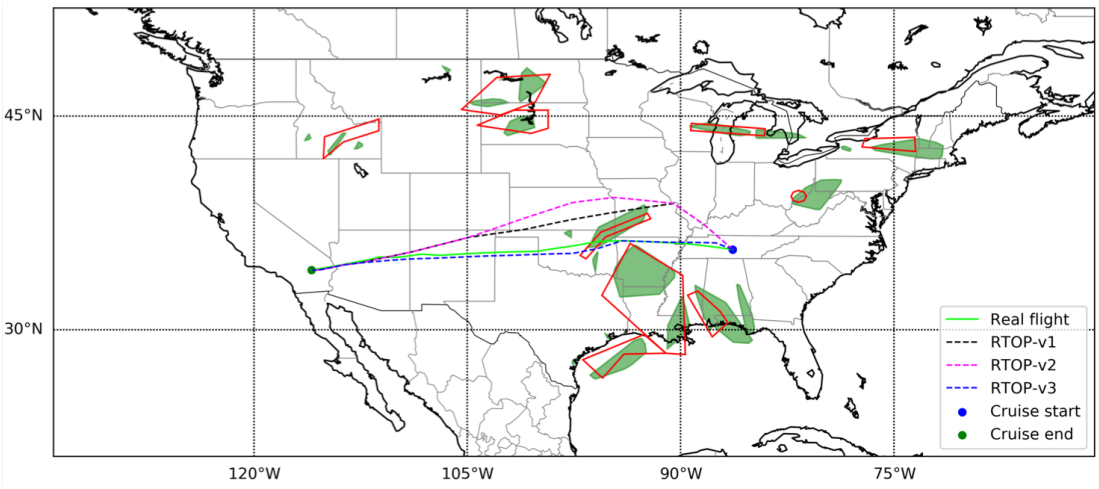
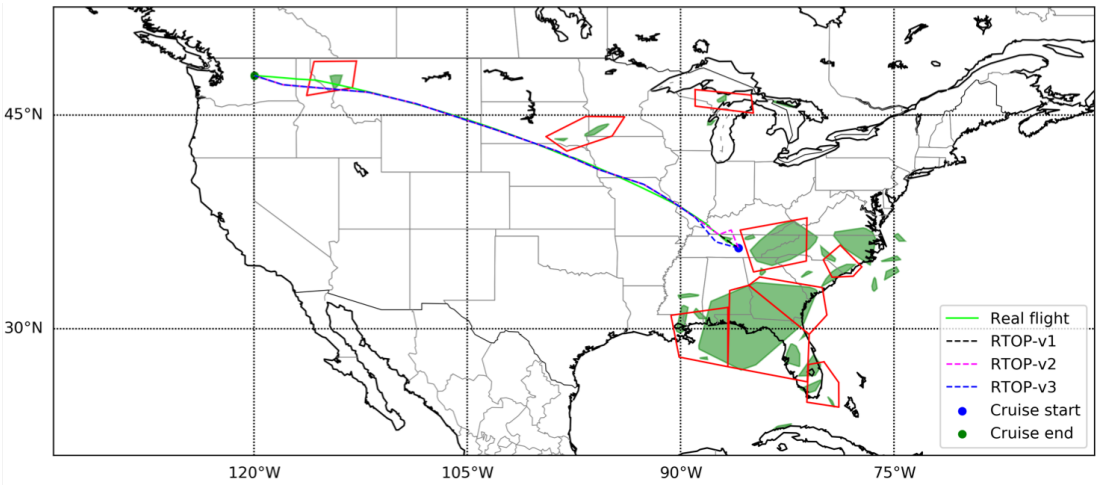


Figure 6.10: Results of the Hausdorff distance for the representative flight cases

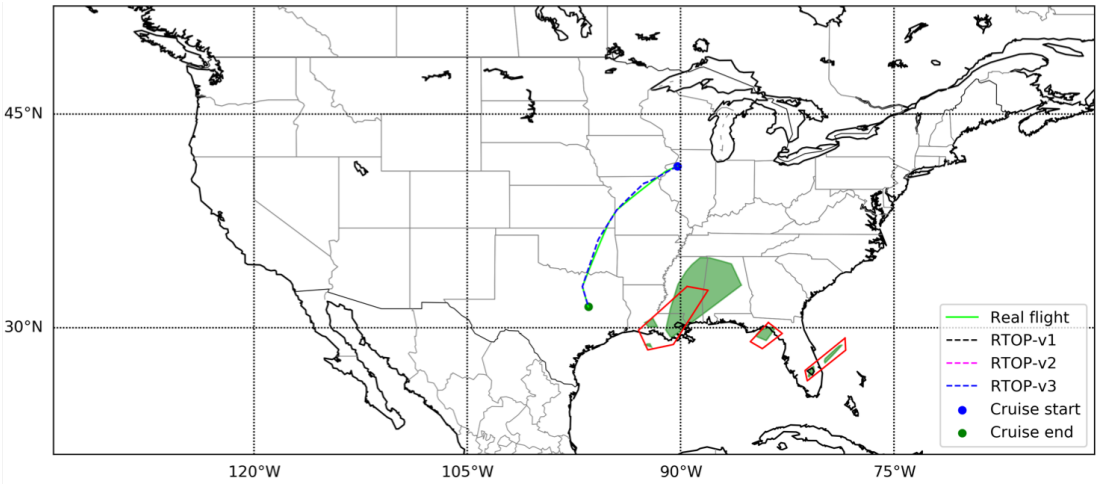
The results indicate the following significant observations: First, it appears that the proposed methodology yielded small Hausdorff distances (i.e., median case) in most cases, meaning that the framework developed in this research would be able to generate simulated flight routes that might be similar to the actual flight routes created by the flight dispatchers of the selected airlines. For example, Figure 6.11 shows the results of some of the median cases indicating that the actual flight trajectory is very similar to the optimal flight trajectory generated by the RTOP-v3.



(a) Simulation results for the ATL–LAX flight



(b) Simulation results for the ATL–SEA flight



(c) Simulation results for the ORD–IAH flight

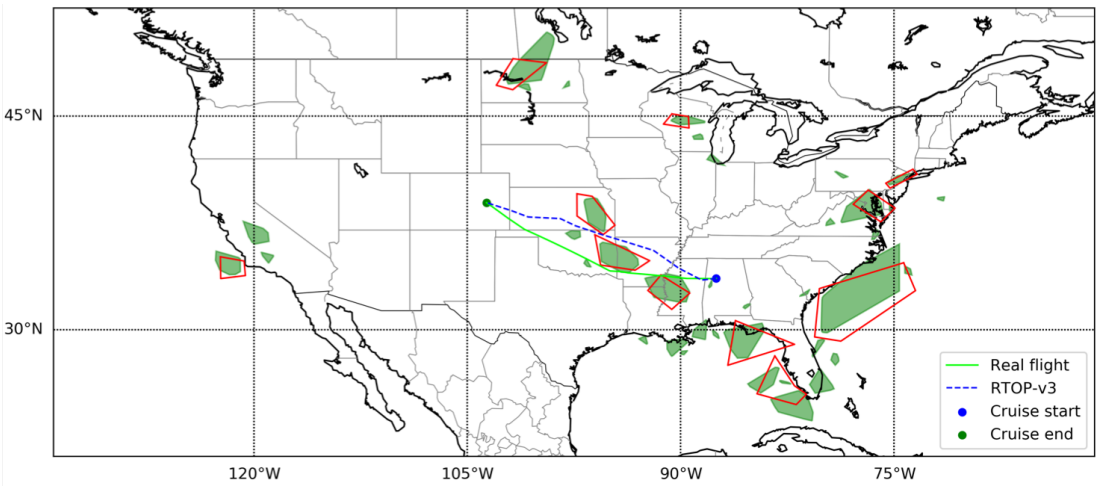
Figure 6.11: Simulation results for the median cases of the Hausdorff distance box plot

The fact that the proposed methodology was able to mimic the behavior of the actual flight trajectories operated by the selected airlines is significant because this implies that the framework developed in this research can be used for small airline operators (e.g., operating private jet flights) that do not necessarily have flight dispatchers but rather ask pilots to manually handle in-flight activities (e.g., generating and updating flight plans). Therefore, it is expected that the framework developed in this research can alleviate the cockpit workload of the pilots employed by small airline operators by providing the capability to continuously optimize flight routes with the latest weather information sets available, potentially leading to increased work efficiency.

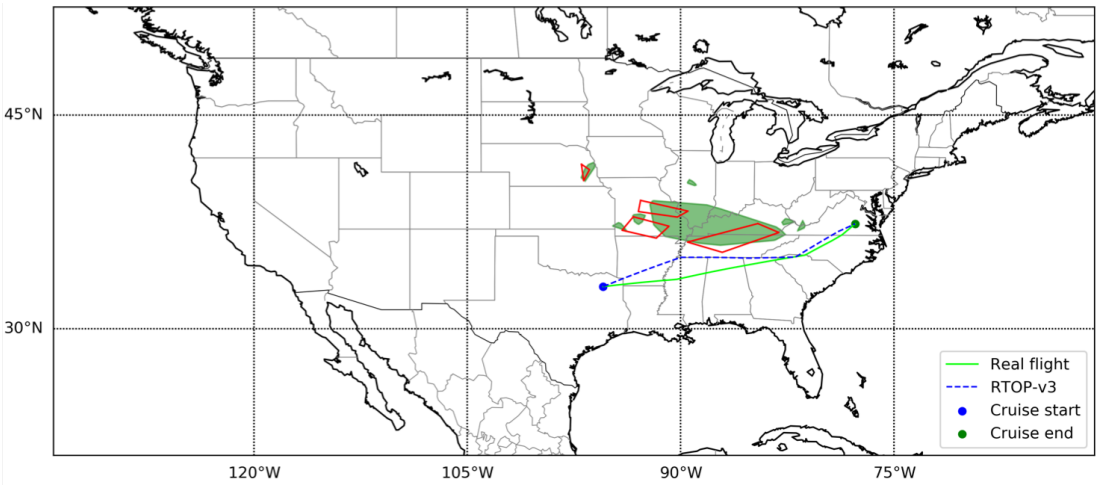
Another significant observation is that the proposed methodology sometimes generated simulated flight trajectories that were significantly different from the real flight trajectories. While these cases are categorized as an outlier in the Hausdorff distance context, it is important to note that the outlier cases do not have a negative meaning (e.g., increasing travel time compared to a real flight case) from the perspective of optimization. Figure 6.12 shows the results of some of the outlier cases indicating that the two trajectories (i.e., actual vs. RTOP-v3) are significantly different.

In particular, as shown in Figure 6.12a, it seems that the RTOP-v3 made the tactical decision to ultimately reduce travel time while the real flight case flew a detour either due to a lack of weather information or airspace congestion. In other words, the RTOP-v3 generated the simulated flight route more efficiently but less conservatively compared to the real flight route, resulting in increasing the Hausdorff distance; thus, the case was categorized as an outlier in the Hausdorff distance context. It should be noted that this is not related to flight path optimization but more associated with polygon datasets provided to the framework. Presumably, the real flight case in Figure 6.12a conservatively optimized the flight route based on the red polygons generated by the AWC.

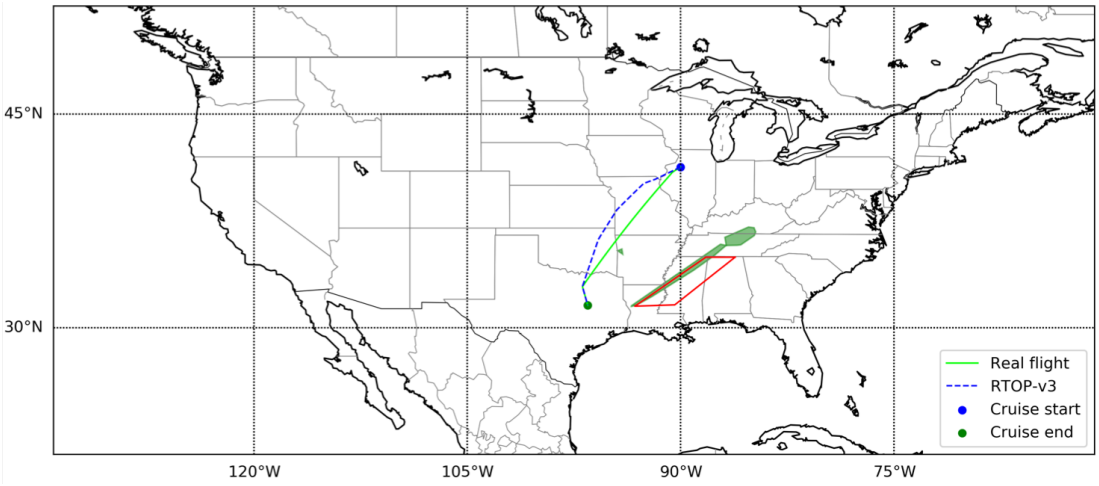
Similarly, Figure 6.13 shows that the RTOP-v3 generated the simulated flight route reducing travel time compared to the actual flight route by making the tactical decision (i.e.,



(a) Simulation results for the ATL–DEN flight



(b) Simulation results for the DFW–DCA flight



(c) Simulation results for the ORD–IAH flight

Figure 6.12: Simulation results for the outlier cases of the Hausdorff distance box plot

optimizing the flight trajectory based on the green polygons generated by the unsupervised machine learning-based convective weather model); therefore, this case was also categorized as an outlier in the Hausdorff distance context. The RTOP-v1, however, generated the simulated flight trajectory similar to the real flight trajectory as it optimized the flight route based on the red polygons, leading to decreasing the Hausdorff distance as shown in Table 6.1. This implies that the framework developed in this research can provide optimal flight routes either tactically optimized by green polygons or conservatively optimized by red polygons.

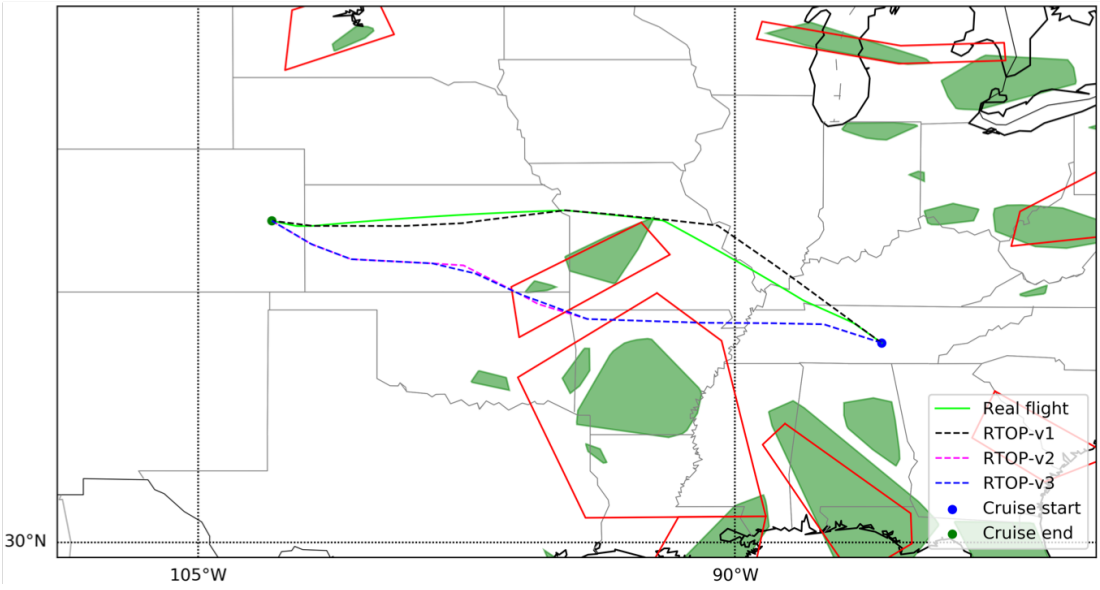


Figure 6.13: Simulation results for the DL1072 flight case

Table 6.1: Travel time and Hausdorff distance comparison between DL1072 and RTOP

Method	Travel time (second)	Hausdorff distance
RTOP-v1	7350	0.61
RTOP-v2	7187	2.95
RTOP-v3	7163	2.96
DL1072	7230	0.00

In addition to the impact of the areas of convective weather activity during the flight path optimization process in the framework, Figure 6.12c shows how the proposed methodology finds an optimal flight path by taking advantage of favorable winds (i.e., riding tailwinds but avoiding headwinds). More specifically, as shown in Figure 6.14, it seems that the RTOP-v3 generated the simulated flight trajectory (6,122 seconds) better than the real flight case (6,290 seconds) in a way that the objective function (i.e., minimizing travel time) would be optimized by seeking favorable winds along with the trajectory.

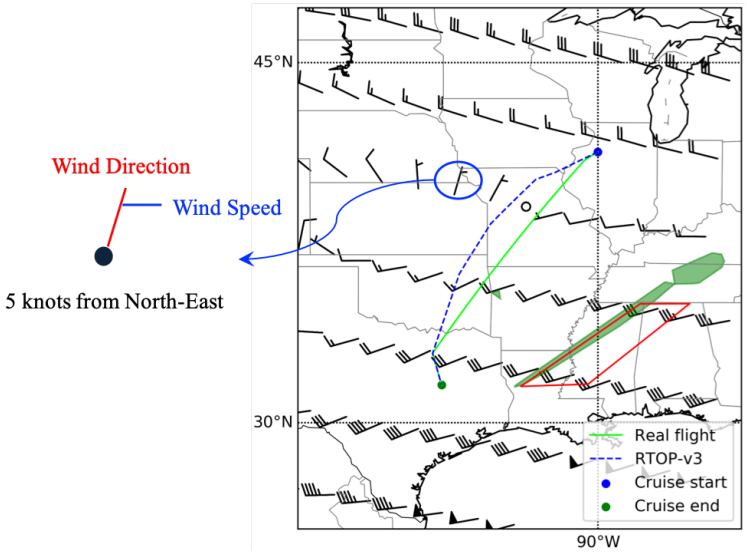


Figure 6.14: Simulation results for the UAL1872 flight case

Figure 6.15 shows the distribution of the potential benefits for all of the representative flights used for statistical analysis. The x-axis in the distribution plot represents the percentage of change in travel time of the simulated flight trajectory compared to the real flight trajectory. The y-axis in the plot refers to the number of the representative flights. The results indicate that the framework developed in this research generated the simulated flight routes that reduced flight time (i.e., en-route phase only) by up to two percent in most cases.

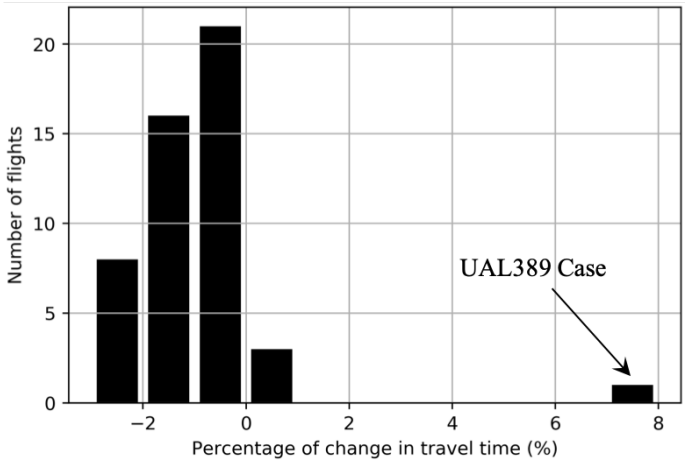


Figure 6.15: Distribution of the potential benefits for all of the representative flights

While it is remarkable that the proposed methodology generated the optimal flight routes that reduced the duration of the en-route phase compared to the real-world flight routes in most cases, it is worth mentioning that the proposed methodology sometimes failed to generate a simulated flight trajectory reducing flight time compared to a real-world flight operation.

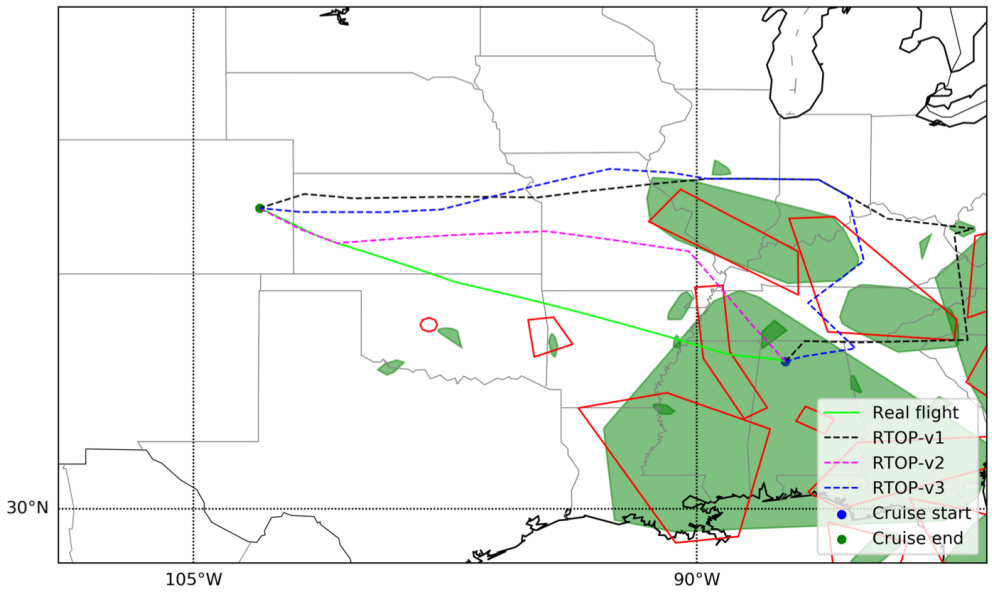


Figure 6.16: Simulation results for the UAL389 flight case

For example, Figure 6.16 shows the results of one particular flight where the RTOP-v3 yielded a flight approximately seven percent longer than the actual flight case (i.e., UAL389) because the RTOP-v3 started to search for optimizing routes while being surrounded by polygon areas of convective weather activity. There is thus an area for future research that could improve the fidelity of the proposed methodology to especially deal with this type of situation.

6.2 Runtime Analysis

Addressing complexity and assured autonomy generally requires 1) Design Time Assurance (DTA) where the system is validated through offline analysis and 2) Run-Time Assurance (RTA) where the system is monitored during live operation to determine if the system is operating correctly [113]. For example, John Schierman et al. [114, 115, 116, 117] developed a framework to evaluate highly adaptive flight control systems. They particularly proposed three different options: 1) all of the processes are performed online, 2) all of the processes are performed offline, and 3) combine the first and second options (e.g., simplified prediction online with high fidelity simulation studies offline).

This research specifically concentrated on the evaluation of DTA through offline analysis with historical real-world flight operations because it was very challenging to conduct a live operational trial of the framework with appropriate members including flight dispatchers, air traffic control coordinators, and pilots. The best way to measure DTA would be to benchmark the framework developed in this research with the other flight planning tools introduced in the literature search such as FACET; however, running the other flight planning tools was challenging due to license issues. For this reason, to discuss the efficiency of the framework, this research particularly considered a tracking metric that shows the speed of the algorithm by plotting elapsed time against origin-destination great circle distance for all of the representative flights.

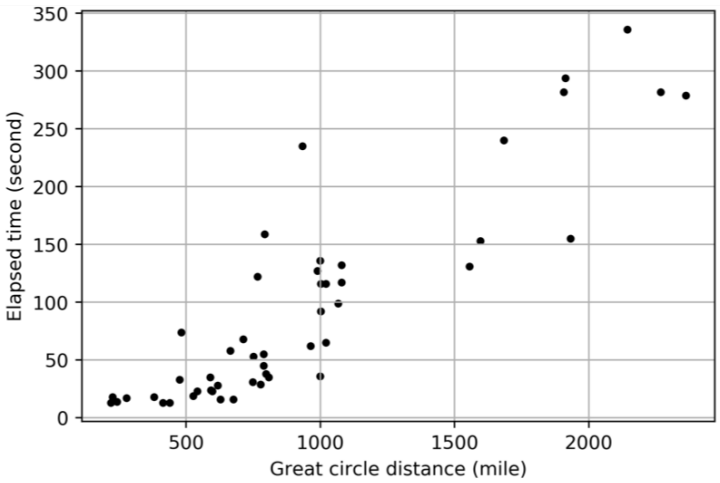


Figure 6.17: O-D great circle distance vs. Elapsed time for representative flight cases

Figure 6.17 shows the results of the tracking metric (i.e., elapsed time vs. origin-destination great circle distance) for all of the representative flight cases. This result indicates the following observations: First, the framework developed in this research generated the optimal flight trajectories in five minutes in most cases, meaning that the framework would be consistently able to create a new flight trajectory (i.e., every five minutes) as needed. Given that it is expected that nothing really happens a lot in five minutes in flight, it is promising that the framework provides the capability to proactively and continuously optimize flight routes if necessary. Second, as expected, the framework required more elapsed time for longer flights.

It must be noted that the results shown in Figure 6.17 do not represent the whole processing time from beginning to the end of the framework. For example, the elapsed time did not include weather data pre-processing steps (e.g., downloading wind data). Figure 6.18 shows a bigger picture of the framework process where both pre-flight (i.e., red color) and in-flight activities are elaborated.

Details of the process are as follows: Step 1 is to download relevant weather datasets. This includes both wind and observational datasets such as radar. Since wind and radar datasets are relatively bigger than the other weather-related datasets, this research identi-

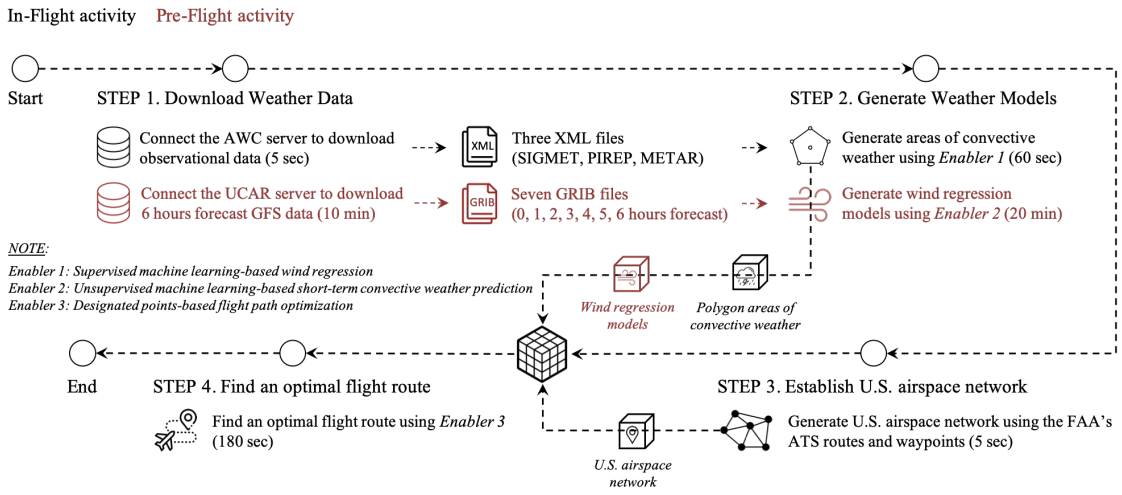


Figure 6.18: Overview of the RTOP-v3 process

fies the steps associated with the two datasets (i.e., wind and radar) as a pre-flight activity. Step 2 is to generate weather models that include wind regression and convective weather models. While the convective weather modeling (i.e., unsupervised machine learning-based short-term convective weather prediction) processing time is typically completed in a minute, generating wind regression models using supervised machine learning algorithms generally requires a high computational cost; thus, the wind regression modeling is identified as a pre-flight activity. Given that the NOAA provides forecast datasets 6 hours ahead of time, this research assumes that the framework is continuously communicating with ground stations where someone trains wind data to receive wind regression models in a timely manner. Step 3 is to establish U.S. airspace network using the FAA's ATS routes and waypoints. Once all of the datasets (i.e., polygon areas of convective weather, wind regression models, U.S. pre-determined airspace network, aircraft true airspeed) are integrated, Step 4 is to find an optimal flight route using the designated points-based flight path optimization model.

The current framework developed in this research in itself does not complete the development of a real-time flight path recommendation system that is fully real-time because of the challenges such as collecting all of the weather datasets in real-time. While the weather

data pre-processing step (e.g., downloading and training wind datasets) is currently considered as one of the most challenging parts for practical consideration in the aviation industry (e.g., commercialization), with the evolution of Integrated Modular Avionics (IMA) architectures and Multi-Core Processors (MCP) [118], it is expected that the implementation of the on-board aircraft equipment systems based on IMA and MCP will significantly reduce computational costs; therefore, the framework developed in this research may be applicable and feasible in real-time analyses in the future once the framework is equipped with a fast aviation connectivity system to enable timely connections to relevant data while in-flight.

CHAPTER 7
CONCLUSION

This chapter concludes this dissertation with a recapitulation of research statements, a summary of contributions, recommendations for future work, and potential applications. The first section revisits the research objective, research questions, research hypotheses, and research experiments formulated for this dissertation and encapsulates the research statements. The second section summarizes the contributions of this research. The third section outlines potential applications based on the outcome of this research. The final section recommends possible topics for future investigation.

7.1 Recapitulation of Research Statements

This research starts by observing two potential issues related to current in-flight re-planning systems presented in Chapter 1. The identified issues are as follows: First, current in-flight re-planning systems are not fully automated; thus, pilots today perform some portions of the in-flight activities manually. Second, weather forecasts used for current in-flight re-planning systems are not always accessible in a timely manner. Therefore, the objective of this research is to develop a framework that automatically performs in-flight re-planning continuously with the latest weather information sets available.

In relation to the research goal, the literature review presented in Chapter 2 outlines several attempts to develop capabilities improving current in-flight re-planning activities and identifies research gaps that need to be bridged. The identified research gaps are as follows: First, a flight management system reduces the workload of the flight crew by automating a variety of in-flight activities but some parts in the flight planning functionality are still manual input. Second, a ground-based flight planning framework has mostly automated the task of flight planning for flight dispatchers but the framework does not address the connec-

tivity issue of co-constructing knowledge between pilots and flight dispatchers, leading to disapproved requests resulting in inefficient communication. Third, the most widely used aviation-related applications implemented in the electronic flight bag have proved their capabilities in certain application areas; however, the applications may not be feasible for a real-time flight path optimization framework that uses the latest weather information to continuously update flight routes. Fourth, deep learning techniques are prevalently adopted in the aviation industry; however, these techniques may not be feasible for a real-time flight path optimization framework due to high computational costs.

The research gaps lead to define an overarching research question: "How can a cockpit-based real-time flight path optimization framework, which automatically performs in-flight re-planning continuously with the latest weather information, be developed?". The overarching research question establishes two requirements raised by two research questions: "How can the capability of providing weather information accurately and continuously to a cockpit-based real-time flight path optimization framework within a specified time be developed?" and "How can the capability of providing simulated optimum flight routes automatically to a cockpit-based real-time flight path optimization framework be developed?".

Chapter 4 presents the formulation of three research problems (i.e., supervised machine learning-based wind regression, unsupervised machine learning-based convective weather prediction, and designated points-based flight path optimization) using the scientific method to answer the two research questions. The literature search on the research problems results in the following observations: First, the most common approach for obtaining continuous wind information in current in-flight re-planning systems is via a linear interpolation method; however, the linear interpolation method may have some limitations in predicting wind patterns where non-linear behavior is dominant. Second, the AWC convective weather data is widely used in the aviation industry; however, the dataset has some limitations in its operations; thus, a graphical representation of the AWC convec-

tive weather data does not always accurately reflect the actual convective weather activity. Third, the A* search algorithm is the most appropriate pathfinding approach for a real-time flight path optimization framework; however, the A* search algorithm finds an optimal flight path only with user-supplied information.

Based on these observations, the following research questions are developed: "How can continuous wind information be provided more accurately (i.e., lower prediction error) other than using the current linear interpolation method?", "How can the areas of convective weather activity be defined more accurately than the AWC convective SIGMET polygon data?", and "How can the constraint of the A* search algorithm (i.e., find an optimal flight path using only pre-determined airways) be relaxed to provide more flight route options that are available to pilots?".

The three research questions lead to construct the following research hypotheses: First, a supervised machine learning-based regression method will provide wind forecasts with a lower prediction error (i.e., RMSE) compared to a linear interpolation method. Second, an unsupervised machine learning-based clustering algorithm with multiple observational datasets which are updated every 10 minutes will define the areas of convective weather activity more accurately than the AWC convective weather data by generating convective weather polygons in a more frequent manner. Third, a hybrid method that combines the A* search algorithm with a free-flight approach will automatically provide flight route opportunities that are not necessarily constrained to established waypoint-to-waypoint airways but rather generate acceptable free-flight route options.

Chapter 5 proposes a data-driven approach that leverages various machine learning and optimization algorithms to test the three research hypotheses with the following research experiments: First, a supervised machine learning-based regression method is compared to a linear interpolation method in terms of testing points randomly sampled in the U.S. territory. Second, an unsupervised machine learning-based short-term convective weather prediction method is visually compared to the AWC convective SIGMET data. Third, the

Hausdorff distances for simulated optimum flight routes generated by both a designated points-based flight path optimization model (i.e., RTOP-v3) and the A* search algorithm (i.e., RTOP-v2) are computed to see which route is more similar to a real flight trajectory, especially under severe weather conditions.

If the aforementioned two requirements are satisfied by demonstrating the three research hypotheses, an overall research hypothesis can be constructed as follows: If a cockpit-based real-time flight path optimization framework developed in this research utilizes 1) accurate wind field datasets predicted by a supervised machine learning-based wind model (i.e., Enabler 1), 2) reliable convective weather polygon areas generated by an unsupervised machine learning-based convective weather model (i.e., Enabler 2), and 3) flight planning functionalities by a designated points-based flight path optimization model (i.e., Enabler 3), then the framework can automatically provide simulated flight route options shorter than real-world flight routes by continuously optimizing the simulated flight trajectories. The overall research hypothesis is substantiated with a set of simulation scenarios (i.e., statistical analysis) presented in Chapter 6.

7.2 Summary of Contributions

The primary contribution of this research is the establishment of an automated framework, especially for commercial airlines, that provides the capability to continuously optimize flight routes using the latest weather information sets available. The framework developed in this study also establishes a basis for optimizing flight routes for all categories of airplanes as well as commercial aircraft. In other words, the outcome of this research can be used for optimizing flight routes of private business jets as needed.

Another contribution of this research is that the automated framework presented in this dissertation leverages a variety of machine learning techniques to improve the fidelity of current in-flight re-planning systems. More specifically, the framework 1) overcomes the limitations of the current convective weather products by implementing an unsuper-

vised machine learning-based short-term convective weather prediction model, 2) provides better wind prediction by utilizing a supervised machine learning-based wind prediction model, and 3) generates optimized flight routes that are not necessarily constrained by user-supplied airway/waypoint information (e.g., FAA pre-determined U.S. airspace infrastructure) by combining a traditional path optimization method (i.e., A* search algorithm) with a free-flight approach (i.e., designated points-based flight path optimization model). Consequently, this dissertation has resulted in several publications as follows:

- Data-Driven Approach using Machine Learning for Real-Time Flight Path Optimization, AIAA Journal of Aerospace Information and Systems, 2021.
- Designated Points-based Free-Flight Approach to Enable Real-Time Flight Path Planning, AIAA Aviation, 2021.
- Supervised Machine Learning-based Wind Prediction to Enable Real-Time Flight Path Planning, AIAA SciTech, 2021.
- A Data-Driven Approach using Machine Learning to Enable Real-time Flight Path Planning, AIAA Aviation, 2020.
- Aircraft Flight Plan Optimization with Dynamic Weather and Airspace Constraints, ICRAT, 2020.
- Aircraft Mission Analysis Enhancement by using Data Science and Machine Learning Techniques, AIAA Aviation, 2019.
- Data-Driven Approach for Understanding the Impact of Weather on Commercial Flight Path, AIAA Aviation, 2019.

7.3 Potential Applications

This section presents potential applications of the outcome of this research. Three primary application areas have been envisioned for the proposed methodology.

First, the most straightforward application is to use the framework developed by this research to optimize flight routes more accurately and frequently than current in-flight re-planning systems. It is expected that the framework can help flight dispatchers at major airlines of the U.S. by providing the capability to re-route flights continuously more frequently. Second, given that small airline operators of the U.S. do not necessarily have flight dispatchers but rather ask pilots to generate and update flight plans, the outcome of this research can be utilized by the pilots to alleviate the cockpit workload by providing the framework as an application on a tablet PC (e.g., iPad). Third, the accomplishment of this research can be extended to international flight operations once datasets (e.g., radar data, waypoint data, airway data, and wind data) are replaced. This will not change the complexity of the framework as it will employ the same methodology proposed in this dissertation.

7.4 Recommendations for Future Work

This dissertation in itself does not complete the development of the cockpit-based real-time flight path optimization framework. There are thus several areas for future research that could further improve the proposed methodology. This section identifies possible agendas for subsequent work which could lead to further improvements on the basis of this study.

One of the limitations of this research is that the framework does not implement an aircraft performance model as it assumes that aircraft true airspeed is constant during the en-route phase of flight. The proposed methodology can be improved by implementing an aircraft performance model with the Base of Aircraft Data [119] that provides theoretical model specifications and related specific datasets (e.g., fuel flow, operating speed, rate of climb). It is expected that the model will simulate the behavior of aircraft.

Another limitation of this research is that the framework does not consider departure and arrival phases but only focuses on improving the en-route phase. While it is true that both departure and arrival phases are generally standardized to ensure flight safety, the

fidelity of the framework can be improved by taking into account an optimization problem in the areas of departure and arrival phases (e.g., departure procedure optimization).

Last, there is room for improvement in some of the assumptions presented in this research. For example, the fidelity of the framework can be extended further by integrating airspace operational constraints such as traffic-related concerns and flight restriction areas (e.g., Temporary Flight Restriction) [120].

Appendices

APPENDIX A
PILOT INTERVIEW TRANSCRIPTION

Interviewer:

When you fly a business jet, do you always follow Instrument Flight Rule (IFR)? If then, how do you choose a route?

Interviewee:

Yes, I do. Usually, the route is chosen by a company because most airlines have their own preferred routes. A flight dispatcher verifies the route that is typically comprised of area navigation (RNAV) waypoints. We negotiate with Air Traffic Control (ATC) during the flight based on what we are getting from the cockpit. We are seeing the window and communicating with ATC to determine either going left or right around the waypoint. One concern is that there may be some mistakes in the communication. So, we may want to reduce this kind of human error.

Interviewer:

Does it happen, when the dispatcher gives you a route, if you are not happy with the option; then you would amend it?

Interviewee:

Yes. A flight dispatcher typically gives a route that abides by Federal Aviation Administration (FAA) regulations. In most cases, we are trying to accept it as much as possible; however, if needed, we amend it; and this is mostly related to the weather. Dispatchers can help us make a decision but eventually, we will make the decision.

Interviewer:

What part of routes do you mostly try to minimize your operating expenditures? Is it an arrival procedure? En-route? Or something related to a departure procedure?

Interviewee:

We don't have any deviation at a departure procedure. There is no room for optimization. So, it happens really quickly. No compromise. In terms of an arrival procedure, it depends on volumes. Even though we want to go the shortcut, it may not be possible because of the volume; instead, we may have to make a big detour. So, an en-route option has more opportunities to try to find out the shortest route.

Interviewer:

When you are en-route, how often do you change your route while flying?

Interviewee:

As far as fuel concerns, we pretty much look at our fuel desk in our system. Once we get new route information along with wind aloft information from a dispatcher, we just input them on our Flight Management System (FMS) system. If then, a new route is automatically updated with respect to an Estimated Time of Arrival (ETA) and fuel. For an hour and a half flight, we probably do it every 20 minutes. This is a sort of one of the manual procedures that we have to keep doing it continuously.

Interviewer:

When you change your en-route flight route, what factors are the most important?

Interviewee:

I would say that weather is the primary reason. Second, we may consider changing altitudes to avoid the area of high turbulence. Third, we also consider changing altitudes to seek out favorable winds.

Interviewer:

When you are saying about weather, does it mean convective weather? If not, what kind of the weather activities impact your flight trajectory?

Interviewee:

In summer time, it is definitely related to convective activities. In winter time, it is related to snow, icing, and the range of visibility.

Interviewer:

In terms of weather, do you also consider a widespread fog area to avoid?

Interviewee:

We would not necessarily need to avoid it because it is usually gone. Icing and fog are typically not an issue because aircraft is certified whatever types of icing and fog.

Interviewer:

If there is a very huge convective weather area captured along with your flight, would you avoid the entire area?

Interviewee:

We would try to get in the area. We will definitely look at it. We do not try to avoid the very large area entirely. You know, aviation is not black and white. But, in most cases, if we find that there is a very large convective weather area, it has been most likely to be already managed by a flight dispatcher on the ground before departure.

Interviewer:

When it comes to convective weather, is there a way that you rank convective activities? Are all the convective activities same for you? Is there any decision process which may be a binary?

Interviewee:

The rule of thumb is to avoid it by at least 20 miles. Otherwise, it is really hard for us to tilt pitch up and down aircraft. We see the radar system in the cockpit and communicate with Air Traffic Control (ATC) based on the radar information to make a decision. One issue is that we try to characterize convective weather by ourselves such as how tall it is and figuring out can they be broken, which is the reason why we ask our ATC to double-check the status. Another issue that we have is when we are already inside the cloud. We can't really see ahead of itself. In this case, we have to depend on the decision made by ATC. For a regional flight case, one of the decision-making processes is listening to what is going on ahead of you in terms of weather.

Interviewer:

Is this current system still not automated? You make them manually?

Interviewee:

Yes, this is a manual process.

Interviewer:

Can you penetrate in terms of precipitation yellow and red?

Interviewee:

We can penetrate the yellow one, but we try to avoid the red one as much as we could. If we decide to penetrate it, we first look at altitude information. If it is not tall, we can absolutely fly higher.

Interviewer:

Can you penetrate heavy turbulence?

Interviewee:

No, we can't go through heavy turbulence, but we can penetrate moderate turbulence.

Interviewer:

How often your radar data system receives new information?

Interviewee:

The data is actually almost in real-time. The Next Generation Weather Radar (NEXRAD) on the Global Positioning System (GPS) is updated every 15 minutes. One concern is that it does not look like the real radar data. It's the color thing. It's very simple. Red is bad and green is good. It's just a matter of painting. It may not be super useful for a decision-making process.

Interviewer:

Can you see Pilot Report (PIREP) on your Flight Management System (FMS)?

Interviewee:

We can't see PIREP on our machine, but PIREP is typically announced by Air Traffic Control (ATC). Sometimes, a dispatcher lets us know the PIREP record that is obtained from the previous airlines.

Interviewer:

When you are in flight, is it mandatory to report PIREP?

Interviewee:

It is not mandatory but highly encouraged because it is useful.

Interviewer:

When you plan a route and need to decide which waypoint you should go to, presumably wind aloft is the major driving factor to your planning?

Interviewee:

Yes, we input our en-route plans on Flight Management System (FMS), and it gives us an Estimated Time of Arrival (ETA). We don't want to have our flight either too fast or slow. So, the wind is the most important factor for this concern.

Interviewer:

Is this also still not automated? You insert them manually?

Interviewee:

Yes, we should input them by ourselves.

Interviewer:

Where do you get wind aloft information?

Interviewee:

We receive wind information from World Area Forecast System (WAFS). We recognize that there must be uncertainty in the prediction system, but we have seen that the accuracy is pretty much high based on our experiences.

Interviewer:

As far as I know, the numerical weather forecast system does not provide continuous wind information. If you need to know wind information at a particular altitude or waypoint, how would you get the information?

Interviewee:

An interpolation method is used to obtain wind information at any altitude and waypoint. We do not use the technique, but it is given by down-link. The reason why they use the interpolation method? Surprisingly, the current computer system in the cockpit (e.g., Boeing 777) is like a Disk Operating System (DOS), that is a very low-level system compared to the current computer technologies.

Interviewer:

You don't always necessarily flight every waypoint in your airspace. Instead, you can direct to many other different spaces, which may be better than the waypoint. Where that happens, is this really depending on what Air Traffic Control (ATC) envisions along the route? This question is important from the design standpoint. If you speak about these things to someone else, how would you say?

Interviewee:

When that happens, the major consideration is traffic congestion. The secondary consideration is the weather. If there is a hazardous weather area, then we probably get a vector of the amount of the weather and change our flight direction, which is not necessarily to the airport.

APPENDIX B

WIND MEASUREMENT DATA BY WEATHER BALLOONS

Table B.1: Wind measurement data by the NOAA's weather balloons at 2021-02-02 12:00 UTC (Altitude = 250 hPa)

State	Station ID	Latitude	Longitude	Direction (°)	Speed (m/s)
FL	USM00072201	24.5531	-81.7886	284	47.4
FL	USM00072202	25.7544	-80.3831	280	55.0
SC	USM00072208	32.8950	-80.0275	321	19.9
FL	USM00072214	30.4461	-84.2994	318	69.5
GA	USM00072215	33.3558	-84.5672	321	66.1
AL	USM00072230	33.1789	-86.7822	319	62.9
LA	USM00072233	30.3369	-89.8250	316	66.5
MS	USM00072235	32.3189	-90.0800	320	63.1
LA	USM00072240	30.1253	-93.2161	312	52.3
LA	USM00072248	32.4511	-93.8414	309	59.1
TX	USM00072249	32.8350	-97.2986	302	51.3
TX	USM00072251	27.7789	-97.5056	313	36.6
TX	USM00072261	29.3744	-100.9183	296	33.9
TX	USM00072265	31.9425	-102.1892	294	39.6
AZ	USM00072274	32.2278	-110.9558	254	37.0
CA	USM00072293	32.8333	-117.1167	240	40.2
NC	USM00072305	34.7761	-76.8767	237	11.2
NC	USM00072317	36.0981	-79.9428	296	16.7
VA	USM00072318	37.2039	-80.4142	305	15.9

TN	USM00072327	36.2469	-86.5617	322	59.0
AR	USM00072340	34.8350	-92.2594	316	59.1
OK	USM00072357	35.1808	-97.4378	300	54.0
TX	USM00072363	35.2331	-101.7092	288	47.1
TX	USM00072364	31.8728	-106.6981	269	37.4
NM	USM00072365	35.0378	-106.6219	267	39.4
AZ	USM00072376	35.2300	-111.8217	255	38.5
NV	USM00072388	36.0500	-115.1833	245	41.6
CA	USM00072393	34.7500	-120.5667	235	54.5
VA	USM00072402	37.9333	-75.4833	200	9.3
VA	USM00072403	38.9767	-77.4858	280	4.6
OH	USM00072426	39.4214	-83.8217	333	27.6
MO	USM00072440	37.2347	-93.4014	317	49.7
KS	USM00072451	37.7614	-99.9686	294	44.9
CO	USM00072469	39.7675	-104.8694	275	39.6
CO	USM00072476	39.1200	-108.5250	268	39.6
CA	USM00072493	37.7444	-122.2236	226	49.6
NY	USM00072501	40.8650	-72.8628	181	37.4
NY	USM00072518	42.6919	-73.8322	177	23.5
PA	USM00072520	40.5317	-80.2172	321	8.0
NE	USM00072558	41.3200	-96.3669	323	41.6
NE	USM00072562	41.1328	-100.7000	304	30.6
UT	USM00072572	40.7722	-111.9553	252	36.0
NV	USM00072582	40.8600	-115.7422	242	37.9
MI	USM00072632	42.6989	-83.4714	329	9.6
MI	USM00072634	44.9075	-84.7189	319	11.1
WI	USM00072645	44.4986	-88.1119	329	41.0

MN	USM00072649	44.8497	-93.5647	320	45.8
SD	USM00072659	45.4556	-98.4133	310	41.5
SD	USM00072662	44.0728	-103.2100	297	34.8
WY	USM00072672	43.0647	-108.4767	251	23.2
ID	USM00072681	43.5672	-116.2114	235	47.3
OR	USM00072694	44.9092	-123.0083	230	45.3
ME	USM00072712	46.8683	-68.0136	214	66.0
MN	USM00072747	48.5647	-93.3975	325	46.8
ND	USM00072764	46.7717	-100.7594	297	41.0
MT	USM00072768	48.2067	-106.6256	275	48.3
MT	USM00072776	47.4614	-111.3847	248	42.0
WA	USM00072786	47.6806	-117.6267	228	48.6
WA	USM00072797	47.9339	-124.5603	220	19.3
ME	USM00074389	43.8925	-70.2572	190	62.1
IA	USM00074455	41.6114	-90.5817	330	54.5
MA	USM00074494	41.6569	-69.9589	184	63.0
IL	USM00074560	40.1517	-89.3383	331	51.0
FL	USM00074794	28.4667	-80.5500	295	50.9

APPENDIX C

NEXRAD SITE INFORMATION (CONTIGUOUS UNITED STATES)

Table C.1: NEXRAD site location information within contiguous U.S.

Radar Site Name	Latitude	Longitude
KABR	45.45583	-98.41306
KABX	35.14972	-106.82333
KAKQ	36.98389	-77.0075
KAMA	35.23333	-101.70889
KAMX	25.61056	-80.41306
KAPX	44.90722	-84.71972
KARX	43.82278	-91.19111
KBBX	39.49611	-121.63167
KBGM	42.19972	-75.985
KBHX	40.49833	-124.29194
KBIS	46.77083	-100.76028
KBLX	45.85389	-108.60611
KBMX	33.17194	-86.76972
KBOX	41.95583	-71.1375
KBRO	25.91556	-97.41861
KBUF	42.94861	-78.73694
KBYX	24.59694	-81.70333
KCAE	33.94861	-81.11861
KCBW	46.03917	-67.80694
KCCX	40.92306	-78.00389

KCLE	41.41306	-81.86
KCLX	32.65556	-81.04222
KCRP	27.78389	-97.51083
KCXX	44.51111	-73.16639
KCYS	41.15194	-104.80611
KDAX	38.50111	-121.67667
KDDC	37.76083	-99.96833
KDFX	29.2725	-100.28028
KDGX	32.28	-89.98444
KDIX	39.94694	-74.41111
KDLH	46.83694	-92.20972
KDMX	41.73111	-93.72278
KDOX	38.82556	-75.44
KDTX	42.69972	-83.47167
KDVN	41.61167	-90.58083
KDYX	32.53833	-99.25417
KEAX	38.81028	-94.26417
KEMX	31.89361	-110.63028
KENX	42.58639	-74.06444
KEOX	31.46028	-85.45944
KEPZ	31.87306	-106.6975
KESX	35.70111	-114.89139
KEVX	30.56417	-85.92139
KEWX	29.70361	-98.02806
KEYX	35.09778	-117.56
KFCX	37.02417	-80.27417
KFDR	34.36222	-98.97611

KFDX	34.63528	-103.62944
KFFC	33.36333	-84.56583
KFSD	43.58778	-96.72889
KFSX	34.57444	-111.19833
KFTG	39.78667	-104.54528
KFWS	32.57278	-97.30278
KGGW	48.20639	-106.62417
KGJX	39.06222	-108.21306
KGLD	39.36694	-101.7
KGRB	44.49833	-88.11111
KGRK	30.72167	-97.38278
KGRR	42.89389	-85.54472
KGSP	34.88306	-82.22028
KGWX	33.89667	-88.32889
KGYX	43.89139	-70.25694
KHDX	33.07639	-106.12222
KHGX	29.47194	-95.07889
KHNX	36.31417	-119.63111
KHPX	36.73667	-87.285
KHTX	34.93056	-86.08361
KICT	37.65444	-97.4425
KICX	37.59083	-112.86222
KILN	39.42028	-83.82167
KILX	40.15056	-89.33667
KIND	39.7075	-86.28028
KIWA	33.28917	-111.66917
KIWX	41.40861	-85.7

KJAX	30.48444	-81.70194
KJGX	32.675	-83.35111
KJKL	37.59083	-83.31306
KLBB	33.65417	-101.81361
KLCH	30.125	-93.21583
KLGX	47.1158	-124.1069
KLIX	30.33667	-89.82528
KLNX	41.95778	-100.57583
KLOT	41.60444	-88.08472
KLRX	40.73972	-116.80278
KLSX	38.69889	-90.68278
KLTX	33.98917	-78.42917
KLVX	37.97528	-85.94389
KLWX	38.97628	-77.48751
KLZK	34.83639	-92.26194
KMAF	31.94333	-102.18889
KMAX	42.08111	-122.71611
KMBX	48.3925	-100.86444
KMHX	34.77583	-76.87639
KMKX	42.96778	-88.55056
KMLB	28.11306	-80.65444
KMOB	30.67944	-88.23972
KMPX	44.84889	-93.56528
KMQT	46.53111	-87.54833
KMRX	36.16833	-83.40194
KMSX	47.04111	-113.98611
KMTX	41.26278	-112.44694

KMUX	37.15528	-121.8975
KMVX	47.52806	-97.325
KMXX	32.53667	-85.78972
KNKX	32.91889	-117.04194
KNQA	35.34472	-89.87333
KOAX	41.32028	-96.36639
KOHX	36.24722	-86.5625
KOKX	40.86556	-72.86444
KOTX	47.68056	-117.62583
KPAH	37.06833	-88.77194
KPBZ	40.53167	-80.21833
KPDT	45.69056	-118.85278
KPOE	31.15528	-92.97583
KPUX	38.45944	-104.18139
KRAX	35.66528	-78.49
KRGX	39.75417	-119.46111
KRIW	43.06611	-108.47667
KRLX	38.31194	-81.72389
KRTX	45.715	-122.96417
KSFX	43.10583	-112.68528
KSGF	37.23528	-93.40028
KSHV	32.45056	-93.84111
KSJT	31.37111	-100.49222
KSOX	33.81778	-117.635
KSRX	35.29056	-94.36167
KTBW	27.70528	-82.40194
KTFX	47.45972	-111.38444

KTLH	30.3975	-84.32889
KTLX	35.33306	-97.2775
KTWX	38.99694	-96.2325
KTYX	43.75583	-75.68
KUDX	44.125	-102.82944
KUEX	40.32083	-98.44167
KVAX	30.89	-83.00194
KVBX	34.83806	-120.39583
KVNX	36.74083	-98.1275
KVTX	34.41167	-119.17861
KVWX	38.26	-87.7247
KYUX	32.49528	-114.65583

REFERENCES

[1] T. P. Velavan and C. G. Meyer, "The COVID-19 epidemic," *Tropical medicine & international health*, vol. 25, no. 3, p. 278, 2020.

[2] A. T. Bureau, "Effects of Novel Coronavirus (COVID-19) on Civil Aviation: Economic Impact Analysis," *International Civil Aviation Organization (ICAO), Montréal, Canada*, 2020.

[3] European Union Aviation Safety Agency, "Aviation Health Safety Protocol," *Operational Guidelines for the management of air passengers and aviation personnel in relation to the COVID-19 pandemic. Issue*, no. 2, 19.

[4] Federal Aviation Administration, *FAA Forecast Fiscal Years 2019-2039*. U.S. Department of Transportation, 2019.

[5] Flightradar24, *Flight Tracking Statitics*, https://www.flightradar24.com/data/statistics?fbclid=iwar36mduktllq2-dmrg-phnpl67x_kwt_utgixfdsfgb0wqrnskulyqe2mrs, Retrieved: 2021-04-15.

[6] Federal Aviation Administration, *Air Traffic By The Numbers*. U.S. Department of Transportation, 2019.

[7] FAA, *Weather delay*, https://www.faa.gov/nextgen/programs/weather/faq, Retrieved: 2021-04-15.

[8] U.S. Department of Transportation, *What the cost of airline fuel means to you*, https://www.transportation.gov/administrations/assistant-secretary-research-and-technology/what-cost-airline-fuel-means-you, Retrieved: 2021-04-15.

[9] J. Kim, S. Kim, K. Song, Y. Yi, and D. N. Mavris, "Aircraft Mission Analysis Enhancement by Using Data Science and Machine Learning Techniques," in *AIAA Aviation 2019 Forum*, 2019, p. 3311.

[10] M. Sauer, M. Steiner, R. D. Sharman, J. O. Pinto, and D. R. Adriaansen, "Flight planning and execution with multiple weather hazards," *Journal of Air Traffic Control*, 2016.

[11] Derek Mayer, *simBrief*, https://www.simbrief.com/home/, Retrieved: 2021-04-15.

[12] U.S. Department of Transportation, "Chapter 2: En Route Operations," Federal Aviation Administration, Tech. Rep.

[13] SkyVector, *Flight planning and aeronautical charts*, https://skyvector.com, Retrieved: 2021-04-15.

[14] SmartSky Networks, *SmartSky 4G LTE*, https://smartskynetworks.com/business-aviation/how-to-buy-inflight-wifi/smartsky-lite/, Retrieved: 2021-04-15.

[15] R. Walter, C. Sptzer, U. Ferrell, and T. Ferrell, "Flight management systems," *The Avionics Handbook*, 2014.

[16] J. Lockwood, "X-plane 11 flight management system," Laminar Research, Tech. Rep., 2017.

[17] D. L. Bashioum, E. E. Carr, and L. Simpson, "Computer flight planning in the north atlantic," *Journal of Aircraft*, vol. 2, no. 4, pp. 337–346, 1965.

[18] M. C. Dorneich, O. Olofinboba, S. Pratt, and T. Feyereisen, "Integration of weather information into the dispatcher pre-flight route selection process," in *Proceedings. The 21st Digital Avionics Systems Conference*, IEEE, vol. 2, 2002, 7A1–7A1.

[19] K. D. Bilimoria, B. Sridhar, S. R. Grabbe, G. B. Chatterji, and K. S. Sheth, "FACET: Future ATM concepts evaluation tool," *Air Traffic Control Quarterly*, vol. 9, no. 1, pp. 1–20, 2001.

[20] D. McNally, K. Sheth, C. Gong, J. Love, C. H. Lee, S. Sahlman, and J. Cheng, "Dynamic weather routes: A weather avoidance system for near-term trajectory-based operations," in *28th International Congress of the Aeronautical Sciences*, 2012, pp. 23–28.

[21] T. A. Lewis, K. A. Burke, M. C. Underwood, and D. J. Wing, "Weather Design Considerations for the TASAR Traffic Aware Planner," in *AIAA Aviation 2019 Forum*, 2019, p. 3616.

[22] J. M. Maris, M. A. Haynes, and D. J. Wing, "Traffic Aware Planner (TAP) Flight Evaluation," in *14th AIAA Aviation Technology, Integration, and Operations Conference*, 2014, p. 2166.

[23] A. Evans and P. Lee, "Using machine-learning to dynamically generate operationally acceptable strategic reroute options," in *13th USA/Europe Air Traffic Management Research and Development Seminar*, 2019.

[24] Honeywell Aerospace, *GoDirect - Your Direct Route to Reliable Services*, 2019.

[25] The Boeing Company, *ForeFlight*, https://foreflight.com, Retrieved: 2021-04-15.

[26] SAAB, *Aerobahn*, https://www.saab.com/products/security/air-traffic-management, Retrieved: 2021-04-15.

[27] I. Emanuilov and O. Dheu, "Flying high for ai perspectives on easa's roadmap for ai in aviation," *Air and Space Law*, vol. 46, no. 1, 2021.

[28] A. D. Evans, P. Lee, and B. Sridhar, "Predicting the operational acceptance of airborne flight reroute requests using data mining," *Transportation Research Part C: Emerging Technologies*, vol. 96, pp. 270–289, 2018.

[29] H. Arneson, A. Bombelli, A. Segarra-Torné, and E. Tse, "Analysis of convective-weather impact on pre-departure routing decisions for flights traveling between fort worth center and new york air center," in *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3593.

[30] Y. Pang, N. Xu, and Y. Liu, "Aircraft trajectory prediction using lstm neural network with embedded convolutional layer," in *11th Annual Conference of the Prognostics and Health Management Society, PHM 2019*, Prognostics and Health Management Society, 2019.

[31] G. Zhu, C. Matthews, P. Wei, M. Lorch, and S. Chakravarty, "En route flight time prediction under convective weather events," in *2018 Aviation Technology, Integration, and Operations Conference*, 2018, p. 3670.

[32] Y. J. Kim, S. Choi, S. Briceno, and D. Mavris, "A deep learning approach to flight delay prediction," in *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, IEEE, 2016, pp. 1–6.

[33] J. Kirk, S. Balasekar, S. Tarazi, M. Hassan, M. Kirby, and D. N. Mavris, "Parametric Real-Time Navigation En-Route," in *AIAA Scitech 2019 Forum*, 2019, p. 0362.

[34] C. Ramee, J. Kim, M. Deguisgnet, C. Justin, S. Briceno, and D. N. Mavris, "Aircraft flight plan optimization with dynamic weather and airspace constraints," in *International Conference for Research in Air Transportation (ICART)*, 2020.

[35] J. Kim, S. Briceno, C. Justin, and D. Mavris, "A Data-Driven Approach using Machine Learning to Enable Real-Time Flight Path Planning," in *AIAA Aviation 2020 Forum*, 2020, p. 2873.

[36] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[37] Wikipedia, *Artificial neuron*, https://en.wikipedia.org/wiki/Artificial_neuron, Retrieved: 2021-04-15.

[38] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[39] M. Minsky and S. Papert, "An introduction to computational geometry," *Cambridge tiass., HIT*, 1969.

[40] Z. Yanling, D. Bimin, and W. Zhanrong, "Analysis and study of perceptron to solve xor problem," in *The 2nd International Workshop on Autonomous Decentralized System, 2002.*, IEEE, 2002, pp. 168–173.

[41] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[42] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989.

[43] Beam lab, *Deep Learning 101 - Part 1: History and Background*, http://beamlab.org/deeplearning/2017/02/23/deep_learning_101_part1.html, Retrieved: 2021-04-15.

[44] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[45] G. Matheron *et al.*, "Splines and kriging; their formal equivalence," *Materials Science*, 1981.

[46] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, Springer, 2003, pp. 63–71.

[47] D. Duvenaud, "The kernel cookbook," *Advice on Covariance functions*, pp. 12–39, 2015.

[48] R. Fletcher, *Practical methods of optimization*. John Wiley & Sons, 2013.

[49] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise.," in *Kdd*, vol. 96, 1996, pp. 226–231.

[50] A. Likas, N. Vlassis, and J. J. Verbeek, "The global k-means clustering algorithm," *Pattern recognition*, vol. 36, no. 2, pp. 451–461, 2003.

[51] G. J. McLachlan and T. Krishnan, *The EM algorithm and extensions*. John Wiley & Sons, 2007, vol. 382.

[52] Wikipedia, *Density-based spatial clustering of application with noise*, https://en.wikipedia.org/wiki/DBSCAN, Retrieved: 2021-04-15.

[53] E. Fix, *Discriminatory analysis: nonparametric discrimination, consistency properties*. USAF School of Aviation Medicine, 1951.

[54] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE transactions on systems, man, and cybernetics*, no. 4, pp. 580–585, 1985.

[55] S. M. Omohundro, *Five balltree construction algorithms*. International Computer Science Institute Berkeley, 1989.

[56] A. K. Dewdney, "Higher-dimensional tree structures," *Journal of Combinatorial Theory, Series B*, vol. 17, no. 2, pp. 160–169, 1974.

[57] Patrick Henry Winston, *MIT Open Courseware - Artificial Intelligence*, https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/lecture-videos/, Retrieved: 2021-04-15.

[58] P. van Emde Boas, R. Kaas, and E. Zijlstra, "Design and implementation of an efficient priority queue," *Mathematical systems theory*, vol. 10, no. 1, pp. 99–127, 1976.

[59] P. Rozas-Larraondo, I. Inza, and J. A. Lozano, "A method for wind speed forecasting in airports based on nonparametric regression," *Weather and Forecasting*, vol. 29, no. 6, pp. 1332–1342, 2014.

[60] M. Ahearn, E. Boeker, S. Gorshkov, A. Hansen, S. Hwang, J. Koopmann, A. Malwitz, G. Noel, C. N. Reherman, D. A. Senzig, *et al.*, "Aviation environmental design tool (aedt) technical manual: Version 2c," United States. Federal Aviation Administration. Office of Environment and Energy, Tech. Rep., 2016.

[61] A. Kapoor, Z. Horvitz, S. Laube, and E. Horvitz, "Airplanes aloft as a sensor network for wind forecasting," in *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, IEEE, 2014, pp. 25–33.

[62] M. A. Mohandes, T. O. Halawani, S. Rehman, and A. A. Hussain, "Support vector machines for wind speed prediction," *Renewable energy*, vol. 29, no. 6, pp. 939–947, 2004.

[63] R. Ak, O. Fink, and E. Zio, "Two machine learning approaches for short-term wind speed time-series prediction," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 8, pp. 1734–1747, 2015.

[64] A. Grover, A. Kapoor, and E. Horvitz, "A deep hybrid model for weather forecasting," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 379–386.

[65] J. E. Evans and E. R. Ducot, "Corridor integrated weather system," *Lincoln Laboratory Journal*, vol. 16, no. 1, p. 59, 2006.

[66] MIT, *CoSPA Live Display*, https://cospa.wx.ll.mit.edu, Retrieved: 2021-04-15.

[67] R. DeLaura, M. Robinson, M. Pawlak, and J. Evans, "Modeling convective weather avoidance in enroute airspace," in *13th Conference on Aviation, Range, and Aerospace Meteorology, AMS, New Orleans, LA*, Citeseer, 2008.

[68] M. Matthews and R. DeLaura, "Assessment and interpretation of en route weather avoidance fields from the convective weather avoidance model," in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2010, p. 9160.

[69] M. Rubnich and R. DeLaura, "An algorithm to identify robust convective weather avoidance polygons in en route airspace," in *10th AIAA aviation technology, integration, and operations (ATIO) conference*, 2010, p. 9164.

[70] The Weather Company, *WSI En-Route Hazards*, https://business.weather.com/products/enroute-hazards, Retrieved: 2021-04-15.

[71] Aviation Weather Center, *National Weather Service*, https://aviationweather.gov, Retrieved: 2021-04-15.

[72] National Oceanic Atmospheric Administration, *National Weather Service Instruction 10-811*, 2019.

[73] H. K. Ng, S. Grabbe, and A. Mukherjee, "Design and evaluation of a dynamic programming flight routing algorithm using the convective weather avoidance model," in *AIAA guidance, navigation, and control conference*, 2009, p. 5862.

[74] L. Blasi, S. Barbato, and M. Mattei, "A particle swarm approach for flight path optimization in a constrained environment," *Aerospace science and technology*, vol. 26, no. 1, pp. 128–137, 2013.

[75] C. Taylor and C. Wanke, "Dynamically generating operationally acceptable route alternatives using simulated annealing," *Air Traffic Control Quarterly*, vol. 20, no. 1, pp. 97–121, 2012.

[76] R. Vivona, D. Karr, and D. Roscoe, "Pattern-based genetic algorithm for airborne conflict resolution," in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2006, p. 6060.

[77] R. J. Szczerba, P. Galkowski, I. S. Glicktein, and N. Ternullo, "Robust algorithm for real-time route planning," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 3, pp. 869–878, 2000.

[78] S. Bokadia and J. Valasek, "Severe weather avoidance using informed heuristic search," *AIAA Paper*, vol. 4232, 2001.

[79] D. I. Knapp, T. Jameson, E. Measure, and A. Butler, "Optimized flight routing based on weather impacts grids," in *13th Conference on Aviation, Range and Aerospace Meteorology*, 2008.

[80] G. D. Fett, "Aircraft route optimization using the a-star algorithm," Air Force Institute of Technology, Tech. Rep., 2014.

[81] C. Schilke and P. Hecker, "Dynamic route optimization based on adverse weather data," *Fourth SESAR Innovation Days*, 2014.

[82] P. Parvu and A. Parvu, "Dynamic star search algorithms for path planning of flight vehicles," in *International Workshop on Numerical Modeling in Aerospace Sciences*, 2013, pp. 193–201.

[83] M. Lindner, J. Rosenow, and H. Fricke, "Dynamically optimized aircraft trajectories affecting the air traffic management," *International Conference on Research in Air Transportation*, 2018.

[84] O. N. Skrypnik, E. E. Netchaev, N. G. Arefyeva, and R. O. Arefyev, "Algorithms for aircraft track optimization in flexible routing," in *ITM Web of Conferences*, EDP Sciences, vol. 30, 2019, p. 03 003.

[85] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[86] Qiao, *Pathfinding*, https://qiao.github.io/PathFinding.js/visual/, Retrieved: 2021-04-15.

[87] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[88] X. Yang, L. Deng, and P. Wei, "Multi-agent autonomous on-demand free flight operations in urban air mobility," in *AIAA Aviation 2019 Forum*, 2019, p. 3520.

[89] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge, "Comparing images using the hausdorff distance," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 15, no. 9, pp. 850–863, 1993.

[90] Aerospace Systems Design Laboratory, "ASDL Ph.D. thesis guideline," *Georgia Institute of Technology*, 2006.

[91] National Oceanic Atmospheric Administration, *NOAA upgrades the U.S. global weather forecast model*, https://www.noaa.gov/media-release/noaa-upgrades-us-global-weather-forecast-model, Retrieved: 2021-04-15.

[92] NOAA, *Global Forecast System (GFS)*, https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs, Retrieved: 2021-04-15.

[93] Pygrib, *Python module for reading and writing GRIB files*, https://jswhit.github.io/pygrib/docs/index.html, Retrieved: 2021-04-15.

[94] National Weather Service, *About our WSR 88-D radar*, https://www.weather.gov/iwx/wsr_88d, Retrieved: 2021-04-15.

[95] Federal Aviation Administration, *Chapter 13: Aviation Weather Services*, https://www.faa.gov/regulations_policies/handbooks_manuals/aviation/phak/media/15_phak_ch13.pdf, Retrieved: 2021-04-15.

[96] FlightAware, *Live flight tracking*, https://flightaware.com, Retrieved: 2021-04-15.

[97] Federal Aviation Administration, *Federal aviation administration's aeronautical data delivery service*, http://dev-faa.opendata.arcgis.com, Retrieved: 2021-04-15.

[98] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[99] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[100] D. Shepard, "A two-dimensional interpolation function for irregularly-spaced data," in *Proceedings of the 1968 23rd ACM national conference*, 1968, pp. 517–524.

[101] I. Durre, R. S. Vose, and D. B. Wuertz, "Overview of the integrated global radiosonde archive," *Journal of Climate*, vol. 19, no. 1, pp. 53–68, 2006.

[102] QATAR Airway, *777 flight crew operations manual*, http://www.737ng.co.uk, Retrieved: 2021-04-15.

[103] J. Kim, S. I. Briceno, C. Y. Justin, and D. Mavris, "Supervised Machine Learning-based Wind Regression to Enable Real-Time Flight Path Planning," in *AIAA SciTech 2021 Forum*, 2021.

[104] C. Veness, *Calculating distance, bearing, and more between Latitude/Longitude points*, https://www.movable-type.co.uk/scripts/latlong.html, Retrieved: 2021-04-15.

[105] S. Gillies, A. Bierbaum, K. Lautaportti, and O. Tonnhofer, "Shapely: Manipulation and analysis of geometric objects," *San Francisco: GitHub*, 2007.

[106] C. H. Deetz, "The lambert conformal conic projection," *US Department of Commerce, US Coast and Geodetic Survey, Special Publication*, no. 47, pp. 1–61, 1918.

[107] J. P. Snyder, *Map projections–A working manual*. U.S. Government Printing Office, 1987, vol. 1395.

[108] Raymond Hettinger and Tim Peters, *Heap queue algorithm (a.k.a. priority queue)*, https://github.com/python/cpython/blob/2.7/Lib/heapq.py, Retrieved: 2021-04-15.

[109] S. J. Corrado, T. G. Puranik, O. J. Pinon, and D. N. Mavris, "Trajectory clustering within the terminal airspace utilizing a weighted distance function," in *Multidisciplinary Digital Publishing Institute Proceedings*, vol. 59, 2020, p. 7.

[110] S. Atev, G. Miller, and N. P. Papanikolopoulos, "Clustering of vehicle trajectories," *IEEE transactions on intelligent transportation systems*, vol. 11, no. 3, pp. 647–657, 2010.

[111] A. M. Churchill and M. Bloem, "Clustering aircraft trajectories on the airport surface," in *Proceedings of the 13th USA/Europe Air Traffic Management Research and Development Seminar, Chicago, IL, USA*, 2019, pp. 10–13.

[112] S. Corrado, T. Puranik, O. P. Fischer, and D. Mavris, "A unified, clustering-based framework for detection of spatial and energy anomalies in trajectories utilizing ads-b data," 2021.

[113] Y. Lim, A. Gardi, R. Sabatini, S. Ramasamy, T. Kistan, N. Ezer, J. Vince, and R. Bolia, "Avionics human-machine interfaces and interactions for manned and unmanned aircraft," *Progress in Aerospace Sciences*, vol. 102, pp. 1–46, 2018.

[114] J. Schierman, D. Ward, B. Dutoi, A. Aiello, J. Berryman, M. DeVore, W. Storm, and J. Wadley, "Run-time verification and validation for safety-critical flight control systems," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 6338.

[115] M. Aiello, J. Berryman, J. Grohs, and J. Schierman, "Run-time assurance for advanced flight-critical control systems," in *AIAA Guidance, Navigation, and Control Conference*, 2010, p. 8041.

[116] J. D. Schierman, M. D. DeVore, N. D. Richards, N. Gandhi, J. K. Cooper, K. R. Horneman, S. Stoller, and S. Smolka, "Runtime assurance framework development for highly adaptive flight control systems," Barron Associates, Inc. Charlottesville, Tech. Rep., 2015.

[117] J. D. Schierman, M. D. DeVore, N. D. Richards, and M. A. Clark, "Runtime assurance for autonomous aerospace systems," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 12, pp. 2205–2217, 2020.

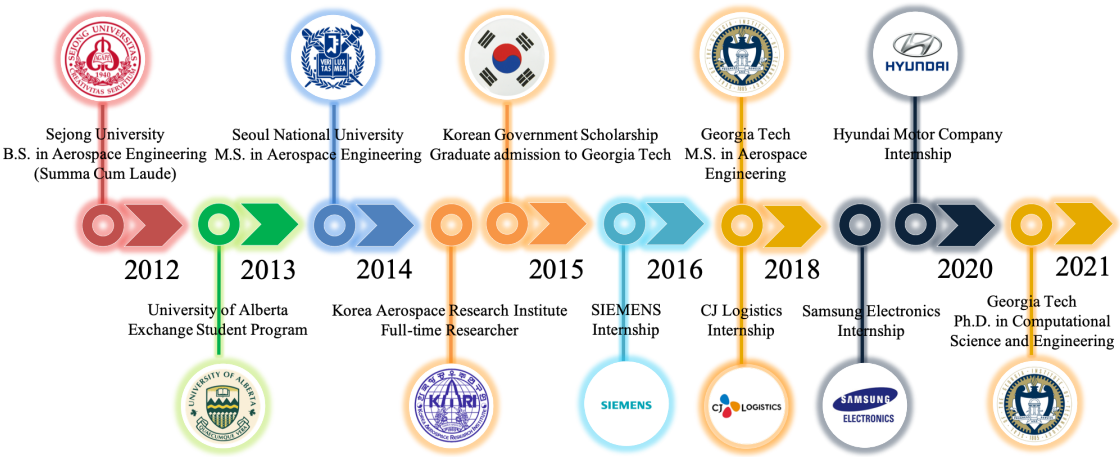
[118] G. Chuyanov, V. Kosyanchuk, N. Selvesyuk, and E. Y. Zybin, "Advanced avionics equipment on the basis of second generation integrated modular avionics," in *29th Congress of the International Council of the Aeronautical Sciences, ICAS*, 2014, p. 2014.

[119] D. Poles, "Base of aircraft data (BADA) aircraft performance modelling report," *EEC Technical/Scientific Report*, vol. 9, 2009.

[120] Federal Aviation Administration, *Temporary Flight Restriction*, https://tfr.faa.gov/tfr2/list.html, Retrieved: 2021-04-15.

VITA

Junghyun Kim was born in Suwon, South Korea. He graduated with honors (Summa Cum Laude) from Sejong University with a B.S. (2012) in Aerospace Engineering. He received his M.S. (2014) in Aerospace Engineering from Seoul National University. He earned his Ph.D. (2021) in Computational Science and Engineering (home unit: Aerospace Engineering) from the Georgia Institute of Technology. He was sponsored by the Korean government for studying abroad (Korean Government Overseas Fellowship) from Fall 2015 to Fall 2017. The following figure provides a visualization of his academic journey to date.

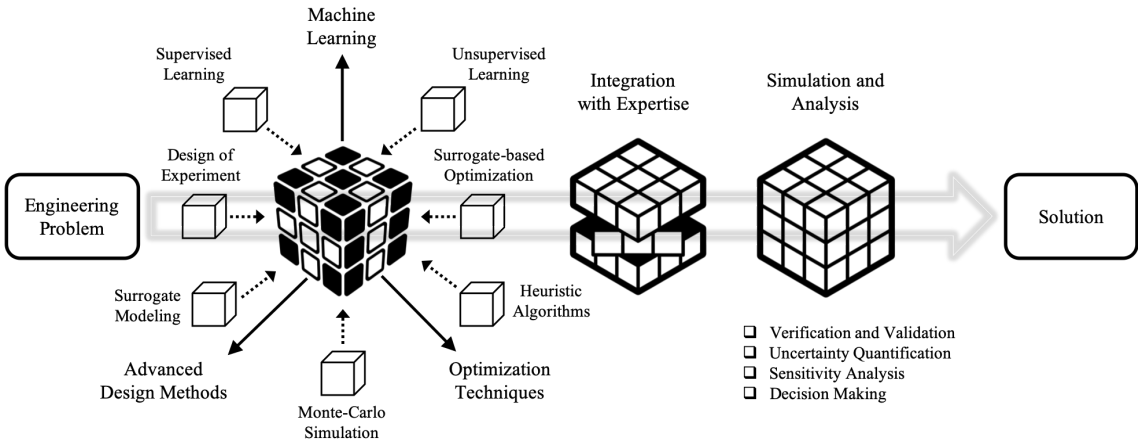


Junghyun Kim's academic journey

During his graduate studies at the Georgia Institute of Technology, he worked at the Aerospace Systems Design Laboratory as a graduate research/teaching assistant. In particular, as a graduate research assistant, he had the opportunity to work closely with the Federal Aviation Administration in order to develop the Aviation Environment Design Tool. He also had a chance to leverage his research skills in various engineering fields while participating in four different internship experiences during his studies at the Georgia Insti-

tute of Technology. Prior to joining the Aerospace Systems Design Laboratory, he worked at the Korea Aerospace Research Institute located in Daejeon, South Korea as a full-time research engineer.

His research focuses on integrating three different areas of specialization (i.e., machine learning, optimization, and advanced design methods) and utilizing them to solve real-world problems in various engineering fields. The following figure presents a visualization of his research areas and interests. During his academic career, he had the opportunity to use his areas of specialization to resolve real-world engineering problems by collaborating with a variety of engineers in different industries, including the Aviation, Logistics, Electronics, and Automobile industries. Such industry experiences had made him aware of the importance of inter-disciplinary research as well as hands-on learning activities.



Junghyun Kim's research areas and interests